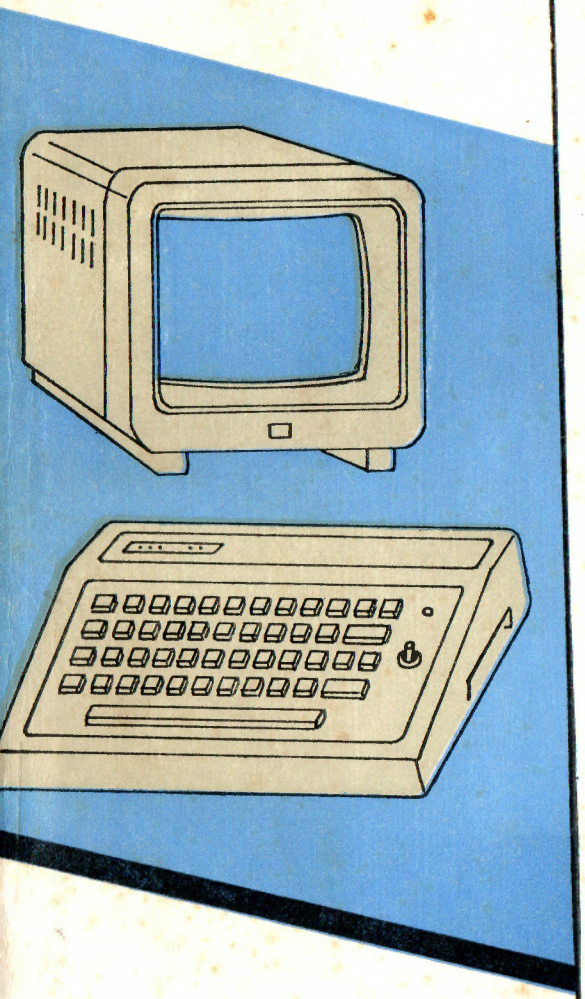
第二课堂丛书

学会使用微型计算机

凌肇元等编著



人民邮电出版社

《第二课堂丛书》编委会

主任委员：茅以升

常务委员：（按姓氏笔划）

王文湛　仇春霖　白金凤　李洛童

陈丽鸣　张太昌　凌肇元

委　　员：（中学部分，按姓氏笔划）

王文湛　仇春霖　甘本祓　白金凤

宁云鹤　朱志尧　刘　涛　安效珍

李三立　李洛童　杨名甲　宋东生

余　杰　陆洪时　陈丽鸣　陈芳烈

张太昌　茅以升　俞锡良　耿文学

徐雄雄　凌肇元　高凤欣　高坦弟

黄寿年

Retro Workshop
果粉工作室

出 版 说 明

本丛书由中国科学技术协会青少年工作部、北京市科普创作协会、科学普及出版社、河北科技出版社、人民邮电出版社联合组织编写。其中中学部分由河北科技出版社、人民邮电出版社出版，小学部分由科学普及出版社出版。

前　言

青少年是世界的未来，国家的希望。在新的世界技术革命的挑战面前，教育只有面向现代化，面向世界，面向未来，才能造就出二十一世纪的一代新人。单纯以课本、课堂和教师传授知识为中心的传统教学方式，已很难使学生更快更广地获取新知识；很难充分地实施因材施教的原则，使每个学生的聪明才智都得到发展；很难培养出成千上万具有创造志向、创造才干和良好科学素质的现代化人才。

学生在上学期间，无疑应该学好教学大纲规定的课堂内容，打下系统而扎实的基础知识，但还要创造条件，更多地运用报刊、广播、课外书籍等来补充新知识，广泛开展形式多样的动手动脑的课外科技活动，通过以实践活动、社会教育、家庭教育和学生自学为中心的“第二课堂”，去获取多方面的知识，锻炼各种能力。这样，课堂学习和课外活动相辅相成，相得益彰，才能培养出具有很强适应能力的，全面发展的，开拓型、创造型人才。

编辑出版这套“第二课堂丛书”，是一种尝试，虽然与“第二课堂”所包含的广阔天地相比，它只是一个小小的枝芽。但它却可以做为一块跳板，引导青少年跃入无限广阔的知识海洋，让他们自己去游泳，去拼搏，破浪前进。

茅以升

一九八五年一月

第二课堂丛书

学会使用微型计算机

凌肇元等 编著

Retro Workshop
果粉工作室

人民邮电出版社

内 容 提 要

这是专门给中小学生和辅导老师、以及其它初接触微型电子计算机的同志编写的学习使用普及型微型计算机的参考读物。为了达到“看了就懂，懂了会用，一用就灵”的目的，书中以教育部推荐的COMX-35型机为重点，结合LASER-310、R-1及APPLE-II等机型，用通俗的语言，介绍了普及型微型计算机的使用方法及简易BASIC语言，并给出若干程序设计实例。

第二课堂丛书
学会使用微型计算机
凌肇元等 编著
责任编辑：沈成衡

*

人民邮电出版社出版
北京东长安街27号
天津新华印刷一厂印刷
新华书店北京发行所发行
各地新华书店经售

*

开本：787×1092 1/32　1985年5月第一版
印张：6 12/32 页数：102　1985年5月天津第一次印刷
字数：142千字　印数：1—130,000册
统一书号：15045·总3043—无6334
定价：0.95元

作者的话

如果你想学会使用微型电子计算机，那么请你读一读这本书。同时建议你读一段，实践一段，边读边做，手脑并用。那么，当你读完这本书的时候，你就很自然地学会了操作简单微型电子计算机了。假如你稍为费点力气，用心把计算机的基本语言即BASIC语言背熟、记住，你就会使自己从一个门外汉步入微型计算机使用者的行列，初步学会了使用简单的微型计算机。所以，我们把这本书叫做“学会使用微型计算机。”

本书的读者不需要具备专门的电子计算机知识，即使是没有接触过电子计算机的小学生、中学生，从事任何工作的职工，只要你想学，就可以从读这本书入门，一步一步地掌握微型计算机的操作方法。至于那些已经对计算机有了初步知识的读者，读一遍这本书也将是一次系统的温习和提高。

目前国内各省市中小学师生和广大职工使用的普及型微型电子计算机品种繁多，但最常见的是下列几种：R-1型(LAMBDA-8300)，比较适用于小学作入门教学；LASER-310型和COMX-35型，比较适用于中学以及一般职工，并可作许多实际应用；HKC-8800型和APPLE-II型则不仅适用于中学、大学作教学机，而且可用于不少不太复杂的实际应用场合。虽然普及型机的型号繁多，具体功能也不一，但它们的基本部分则具有共性。只要掌握其中一种，便可举一反三，触类旁通，无需一种一种从头学起。其中COMX-35型机是教育部向中小学推荐的一种型号，在普及型计算机中属于上下可兼顾的中等水平

机，所以本书选COMX-35型机为重点，并联系到其它几种常用机型号。

本书由凌肇元、韩勐、于致礼同志共同编写，由凌肇元同志主编，并得到南开大学数学系和天津新技术开发公司的大力协助。由于作者水平低、经验少，时间仓促，书中谬误之处必定不少，请广大读者批评指正。

凌肇元

1984年8月

Retro Workshop
果粉工作室

目　录

第一章　微型计算机的基本操作方法……1
1-1　把计算机和电视机连接起来 ……1
1-2　按动键盘，屏幕上将会出现什么……6
1-3　你的计算机会唱歌 ……8
1-4　鲜艳的色彩是这样出现的 ……10
1-5　有趣的图形 ……11
1-6　比计算器高明得多 ……11
第二章　微型计算机概介……13
2-1　微型计算机的基本部件 ……13
2-2　微处理器 ……14
2-3　数据总线、控制总线和地址总线 ……15
2-4　字长和位数 ……16
2-5　运算器和控制器 ……17
2-6　存贮器及其容量 ……18
2-7　只读存贮器(ROM)和随机存贮器(RAM) ……19
2-8　软磁盘和磁带 ……20
2-9　键盘、显示器和打印机 ……22
2-10　程序、语言和软件……23
2-11　微型计算机结构……25
第三章　几种普及型计算机……28
3-1　普及型计算机系统 ……28
3-2　R-1型微型计算机 ……32

3-3 LASER-310微型计算机 …… 33
3-4 COMX-35型微型计算机 …… 35
3-5 HKC8800微型计算机 …… 36
3-6 APPLE-IIPLUS微型计算机系统 …… 39
3-7 普及型计算机的一般用途 …… 41
第四章 程序设计与流程图 …… 44
4-1 指令与程序 …… 44
4-2 流程图 …… 46
4-3 用流程图帮助设计计算机程序 …… 50
第五章 用BASIC语言来设计程序 …… 63
5-1 BASIC语言的常用语句 …… 63
5-2 数字、变量和表达式 …… 72
5-3 运算符号和运算规则 …… 75
5-4 提供数据的语句 …… 78
5-5 数学函数 …… 85
5-6 程序流向控制语句 …… 88
5-7 字符串变量 …… 98
5-8 数组和DIM语句 …… 105
5-9 BASIC语言的另一些命令和语句 …… 111
5-10 用BASIC语言编写和调用机器语言程序 …… 116
第六章 外部设备的使用 …… 122
6-1 显示器 …… 122
6-2 数据记录器 …… 123
6-3 印字机 …… 127
第七章 普及型微计算机的特殊功能 …… 129
7-1 屏幕作图 …… 129
7-2 选择色彩 …… 134

7-3 音响……138
7-4 时间控制功能……143
7-5 屏幕编辑(EDIT)……146
第八章 程序设计实例……152
附录1 Microsoft BASIC语言一览表……174
附录2 字符—代码对照表（参照美国标准信息交换代码ASCII)……179
附录3 错误信息表……184
附录4 普及型微计算机常用名词中英文对照表……189

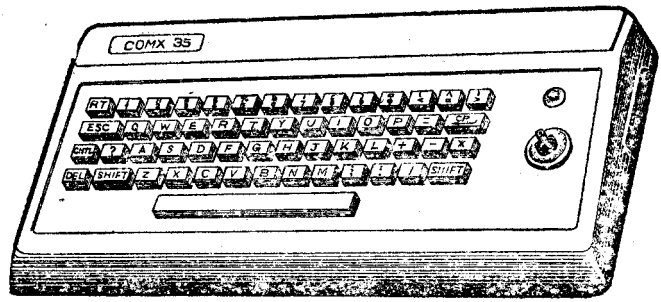
第一章　微型计算机的基本操作方法

1-1　把计算机和电视机连接起来

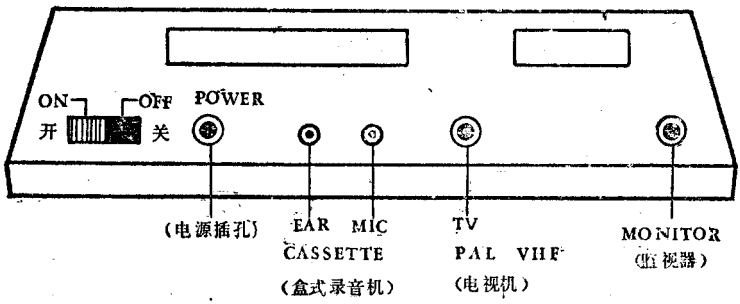
普及型计算机最基本的两件设备是主机和显示器（普及型计算机常用电视接收机）。主机一般还包括电源变压器及与变压器连在一起的电源输入线、一根从主机输出端连到电视机的电缆线、两根从主机输出端连到录音机的连接导线。它们组装在同一个主机包装盒里。显示器不论彩色电视机或黑白电视机均可用，当然用彩色电视机可以获得更好的效果。但即使用23厘米、31厘米或37厘米的家庭用黑白电视机，也同样可以做各种基本操作训练。

假定你还没有接触过电子计算机，那请你先看一看图1·1·1。这是COMX-35型微型计算机主机外形图。其中图（a）是主机正面外形，图（b）是背面的插孔图。背面最左端是电源开关（“ON”表示开，“OFF”表示关）和连接电源变压器的电源插孔（标记“POWER”）。顺着往右看，中间部位有“CASSETTE”字样，表示接盒式录音机。这里有两个插

孔，一个“EAR”和一个“MIC”，其中“EAR”接录音机的耳机插孔，“MIC”接录音机的拾音器插孔。录音机相应的插孔上也标记着“EAR”和“MIC”。再往右看是连接电视



(a) 正面



(b) 背面

图 1·1·1 COMX-35型计算机外形图

机的插孔，标记着“TV”和“PAL/VHF”。此插孔通过专用的电缆线与电视机天线插孔相连。最右端的一个插孔标记有“MONITOR”，是用来和专用的监视器(CRT)相连的。

只要把主机和电视机连接好（见图1·1·2），就可进行计算

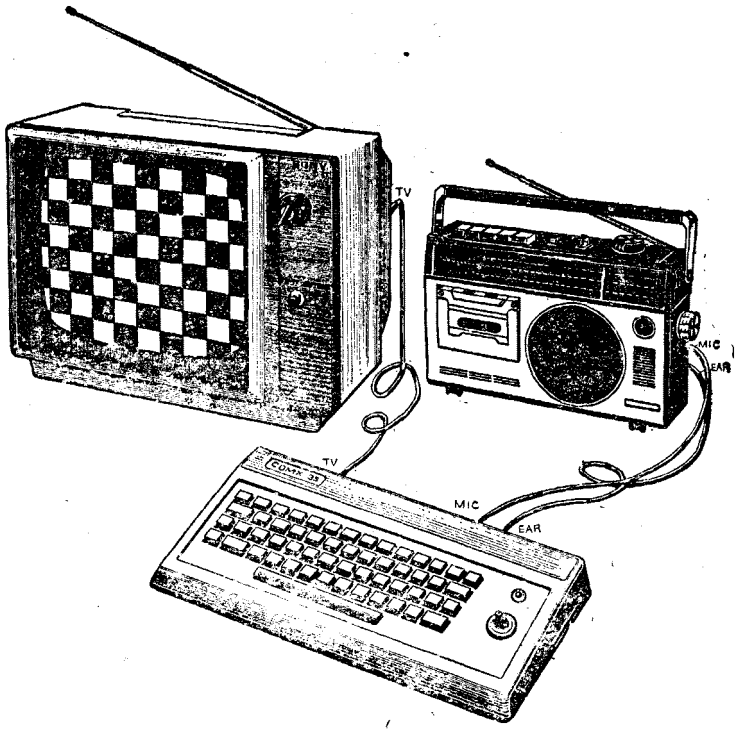


图 1·1·2 COMX-35与电视机录音机连接图

机最基本的操作训练了。连接方法是这样的：从包装盒里取出电视机连接线，一端插在主机背面的“TV”插孔里，另一端和电视机天线插孔相接。如果你手边的电视机是黑白电视机，天线插孔和电缆插头不相配，可另加一对相配的插头、插座，或者加一个随计算机附带来的天线开关盒，见图1·1·3。有了开关盒还可方便地转换收看电视节目。如果天线插孔是双孔式的，那你也可以把电缆线连接天线那端的插头，用电烙铁焊上两根

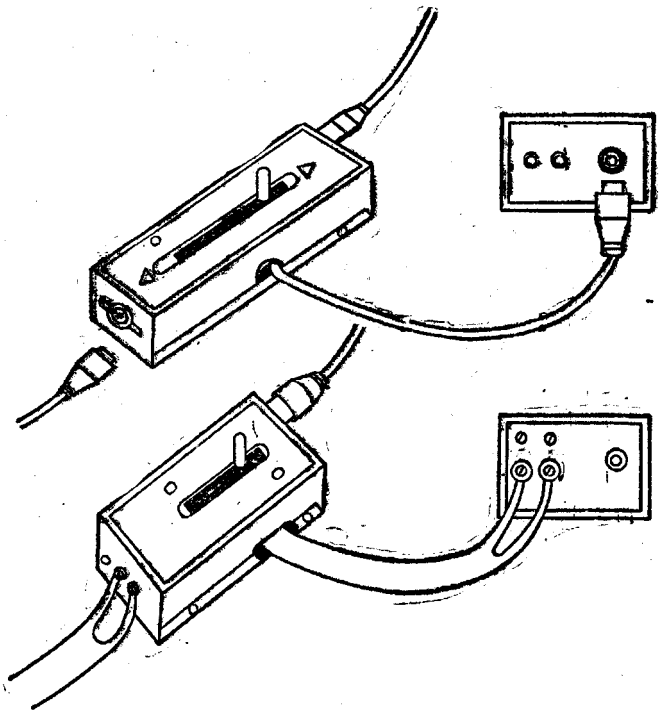


图 1·1·3 开关盒的使用

5厘米长的导线，一根导线焊在中心线上，另一根导线焊在屏蔽线上（即插头的圆柱形金属环上），导线的另外两端经香蕉插头插到电视机天线的两插孔里。当然先要把原来插在插孔中的电视机天线拔掉。如果是彩色电视机，则因机上多已具有和电缆线插头相配的同轴插孔，就可以直接把插头插到天线插孔上了。

把主机所带电源变压器次级的小插头插入主机电源接口(POWER)。初级插到交流220V电源。把主机电源开关断开

（置于“OFF”）。这就做好了准备工作。

计算机与电视机连接时，一定要先关闭计算机和电视机的电源，连好后先打开电视机电源开关，再打开计算机电源开关。

电视机置于VHF频段的二频道，开启电视机，慢慢旋动电视机频道微调钮，使屏幕比较清晰和稳定。再接通主机电源（置于ON），立即可看到主机指示灯（红色发光二极管LED）点亮并听到一支短促的乐曲。这时再微调电视机频率微调钮，并调节亮度和对比度，使屏幕上出现清晰的下列字样

COMX

或出现

COMX

COPYRIGHTED © 1983 BY
COMX
WORLD OPERATIONS LTD

如果你看到电视屏幕上没有字样，只有杂乱无章的图形或者“雪花”，也不要失望，细心地调节一下电视机接收频率，问题便可解决。旋动调谐旋钮要缓慢，亮度和对比度要适中。一般不要去调电视机的帧幅度和帧线性旋钮，也不必去调帧频和行频旋钮，除非电视机原先就没有调好，屏幕上图象不断闪动，才有必要去调帧频和行频的旋钮。

音响发自计算机中的主扬声器，所以电视机的音量要调到最小，以避免噪音干扰。

在屏幕上出现上述字样之后，再按一下除了空格键（即主机面板上最下端的一根长条键）以外的任何键，屏幕上左上角就会出现图1·1·4所示的情况，表示计算机已“准备就绪”，可

以使用BASIC语言（等五章专门介绍）指挥它工作了。

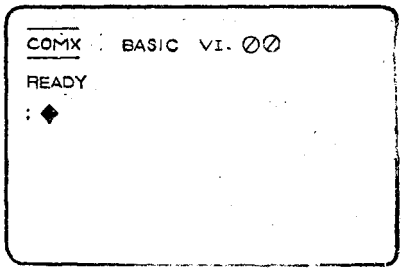


图 1·1·4 屏幕左上角的字样，表示系统已建立

关机时，先将电源开关置于“OFF”位置，再关掉电视机电源，然后拔掉全部电源插头，或者关掉总开关就可以了。

1-2 按动键盘，屏幕上将会出现什么

让我们先来熟悉一下计算机的键盘。首先像上一节介绍的那样接好主机与电视机，使屏幕左上角出现图1·1·4所示的字样。如果屏幕上的字很乱，只要同时按一下RT键（在左上角）和“空格”键（长条键）。RT键表示重新启动，英文为“RESET”。这一操作用以清除原先的程序和数据，相当于关掉电源后再打开电源。

图1·1·4中的冒号“：”，是表明计算机处于BASIC系统的提示符；棱形符号叫做“光标”，在彩色电视屏幕上光标呈粉红色。

整个键盘有55个键，字母键和数码键同英文打字机的排列一致。显然，如果你原来就会使用英文打字机，键盘操作就很容易掌握。

下面让我们来初步实习一下各个键的功能。

1.按一下1键和ϕ键，会看到屏幕上冒号不动，光标右移，冒号与光标间出现1字和ϕ字（键盘上ϕ表示数字0（零）第二排的0表示英文字母O）。再依次按动Q、W、E、R、T各键，屏上冒号与光标间出现并排的一行字“1ϕQWERT”。

假如接着按一下空格键，再依次键入Y、U、I、O、P，屏上将出现“1ϕQWERT YUIOP”。可见，按一下空格键，就空出一格。按两下，就空出二格。

按一下CR键，光标立即转入下一行最左边。CR键叫做“回车”键，由英文“CARRIAGE RETURN”的二个字头组成。

接着按2ϕASDFGHJKL，就会在第二行的冒号与光标间出现上述字样。

再按CR键，则又一次回车，光标移至第三行最左边。

2.键盘第三行最左边为CNTL键，叫做“控制”键，可看作英文CONTROL的缩写。按下CNTL键，屏幕上并不显示。但如果按此键的同时，再按另外的键，会出现一些特殊的功能比如同时按CNTL和I键，光标就向上移一行，并清除上一行中原先的符号。只见上一行的2字被清除，并被光标所代替。

如果同时按下CNTL和K键，光标就右移，并清除相应位置的符号。连续按五次，就把第二行的2ϕASDF清除掉了。

如果同时按下CNTL和M键，光标就向下移动一行，同时按下CNTL和J键，光标就向左移动一行。

3.键盘第四行最左边为DEL键，叫做“清除字符”键，可看成英文为“DELETE”的缩写。按下它，将清除紧挨光标左边的第一个字符。当用键盘打出各种符号时，如果打错了一个符号，就可按一下DEL键，光标就往回移一位，把打错

的符号清除掉。

4.键盘的第一行，不论是数字1、2、3、…，还是<、=、>号，每个键上都有两个符号，第四行的右下部也有类似情况。按下相应的键打出的是该键下部的符号，如果要打出上部的符号，要同时按下该键和SHIFT键（为方便使用，提高打字速度，此键有两个，左右方各一个）。

5.顺便说一下第二行最左边的ESC键。此键叫“中断”键，用于中断执行中的程序，为英文ESCAPE的缩写。

1-3 你的计算机会唱歌

只要你依照一定次序按动主机键盘，计算机里的扬声器便会自动发出乐音，唱起歌来。

你一定很想马上听到计算机发出的音乐吧？好，请依照如下操作次序，按动键盘：[M] [U] [S] [I] [C] [$] （同时按下[SHIFT]）[X] [I] [Y] [I] [Z] [I] （同时按下[SHIFT]键）请注意上面的[X] [Y] [Z]是指某一个数字，不是真的按英文字母键。其中X取1～7范围内的任一个整数，用以选取同一个八度音阶内的不同音符；Y取1～8范围内的任一个整数，用以获得不同的音程；Z取1～15范围内的任一个整数，用以获得强弱不同的音量。众所周知，音阶由频率决定，音程取决于不同的频率段，音量则反映了响度。

例如你打入MUSIC（2，4，5）之后（为方便起见，按键字母或数字外未加方框，下同），再按一下回车键CR，便能听到“RAY”的音响。括号里的数字不同，发出的声音就不同。如果键入MUSIC（ϕ，ϕ，ϕ），则声音消失。

假定你依照如下格式按动相应的按键：

TONE (X, Y, Z)

这里的X取1～128范围内的整数，Y和Z同上，则能听到一个单音，频率决定于X，音程决定于Y，音量决定于Z。

听到单音以后，如果你还想进一步听到一组连续的乐音，则请你按如下次序按动键盘：

```
1ϕ  Z = 3
2ϕ  FOR  Y = 5  TO 7
3ϕ  FOR  X = 1  TO 7
4ϕ  MUSIC (X, Y, Z)
5ϕ  FOR  T = 1  TO  5ϕ : NEXT  T
6ϕ  NEXT  X
7ϕ  NEXT  Y
8ϕ  MUSIC (ϕ, ϕ, ϕ)
9ϕ  END
LIST
RUN
```

当按下LIST时，屏幕上的程序自动排列整齐；当你按下RUN之后，就会听到某调的“1 2 3 4 5 6 7，1 2 3 4 5 6 7，1 2 3 4 5 6 7”的乐音。请注意，键入上述程序的每一行之后，都要按一下CR键（以下同）。直到最后一步，也是在按CR之后才出现音响。

改动一下Z和Y的具体数值，你将体会到当Z加大时，音量会加强；当Y增大时，音程增多。最后你还可改变第5行中T的数值，你将发现当T增大时每个声音的持续时间将加长。这些感性认识，有助于读者今后掌握BASIC语言。

你一定很想了解这个程序是如何制定的。本书后面要介绍

这方面的知识，请你把本书读下去，当你读完之后，就能让计算机唱歌了。

1-4 鲜艳的色彩是这样出现的

运用计算机键盘可以选择多种不同的显示颜色，这只要利用普通的家用彩色电视机，无需专用的显示器。

请你试一试键入如下命令：

SCREEN（5）再按一下CR键，你立即看到整个屏幕的背景呈红色。把括号中的5改成6，整个屏幕的背景就变成黄色。若改成2或3，就出现绿色或蓝色。

再请你试一下，键入如下命令：

COLOR（11）再按一下CR键。你会看到屏幕上键盘输入的符号显示黄色，以后再打入任何符号，均将显示黄色，而计算机响应的符号、数字等则均显示绿色。如果把括号中的11改成4，则键盘输入的符号呈白色，而计算机响应的符号呈深红色。若把括号中的数字再改成别的1～12中某个数字，还会出现其它不同的颜色。

由此可见，电视机屏幕背景的颜色、键盘输入的符号颜色、计算机响应的符号颜色，都可通过键盘输入选取。如果你想了解具体如何选取，有哪些不同的颜色，则请读本书后面的章节，那里有详细的介绍。

如果你想看到屏幕背景八种不同颜色自动变化的情况，不仿按下述次序键入：

```
1φ  REM  COLER  DEMO
2φ  FOR  I=8  TO  1  STEP-1
3φ  SCREEN(I)
```

```
40  FOR J=1 TO 50
50  NEXT J
60  NEXT I
70  END
LIST
RUN
```

1-5 有趣的图形

如果你在按下“SHIFT”键的同时，依次按下O、P、P、Q键，屏幕上就会出现一架飞机的图形，其中SHIFT－O是机首；SHIFT－P是机身，由于机身长，P要按二次（或三次）；SHIFT－Q是机尾。三者连在一起，形成一个飞机图形。

如果在按SHIFT键的同时按K、L、M键，就会得到绿色、蓝色、黑色三种不同颜色的正方块。用一系列的正方块，可以组成某个汉字，或者某种有趣的图形。

这样的图形符号有很多种，如按SHIFT－R可以出现音符图形，按CNTL—D可以出现蓝绿色方框等等，把它们组合到程序中去，可以起各种特殊的作用。

1-6 比计算器高明得多

电子计算机的基本功能当然是运算，和计算器相比，凡是计算器能解决的运算，计算机都能解决，但又比计算器要高明得多。

举个例子来说，根据库仑定律 $F=\frac{Q_1Q_2}{R^2}$ 可计算两个点电

荷之间的吸引力。设两个点电荷$Q_1=2.34$库仑，$Q_2=-0.57$库仑，要求当R＝3时的吸引力F是多少，只要按照命令：

PRINT2.34×0.57/3∧2

键入，便可从屏幕上读出F＝0.1482。

计算机比计算器高明之处在于，如果打入一个程序，就可以求得当R等于任意数时的库仑力F。比如按下述次序键入：

```
1φ  PRINT "R=";
2φ  INPUT  R
3φ  F=2.34*.57/R∧2
4φ  PRINT "F=", F
5φ  END
```

这时屏幕上除显示上述程序外，还出现“READ：”字样，接着键入RUN，屏幕上除即时显示RUN外还出现了“R＝?”。如果你要算R值为3的结果，那么可按下3键，屏幕上马上显示“3”、“F＝0.1482”及“READ：”，告诉你R为3时库仑力F为0.1482，同时还提醒你，计算机已准备好进行R为另外数值时的计算了。如果你还要算R＝2时的F，那么可以再按入RUN。屏幕上又会出现“R＝?”，键入2，屏幕将显示出“F＝0.33345”。这就是说，当你要计算不同R值时的F值的话，每次只要键入RUN和你要取的R值即可。

当然，计算机能解决的运算问题比上述情况要复杂得多。当你读完本书，尤其在熟悉了最后一章所举的程序实例之后，你自己就能运用计算机去解决一系列较为复杂的问题了。

第二章　微型计算机概介

2-1　微型计算机的基本部件

微型计算机是由哪些基本部件组成的呢？以完成一个计算为例，在开始计算前，先要把计算的程序和数据送进计算机里去，所以计算机要有输入设备。其二，每次送进计算机的程序和数据，可能很多很多，必须全部准确记住，以便在计算过程中随时调用；中间计算的结果，也要随时记下来，随时取用。因此，计算机必须有一个记忆装置，这就是存贮器。存贮量的大小，即记忆能力的高低，是衡量计算机质量的主要指标。其三，计算机当然必须会运算，会分析、比较、判断和处理，因此必须要有运算器。其四，算出的结果要告诉人们，或显示或记录等，就还要有输出设备，例如电视机、打印机等都属于输出设备。其五，计算机是按程序自动工作的，执行程序要有个指挥部，以便将各种运算按照一定的顺序进行操作，因此，计算机还要有一个控制器。综上所述，电子计算机是由五个主要部件组成的，即输入设备、存贮器、运算器、输出设备和控制

器，如图2·1·1所示。

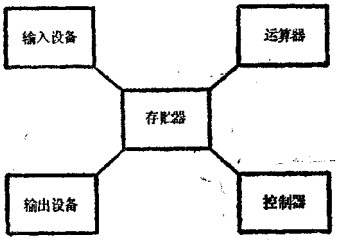


图 2·1·1 微型计算机的五个组成部分

一般把计算机的运算器、控制器和内存贮器合在一起，称为主机。第一章里所说的主机，就是这几部分的组合。外存贮器、输入设备和输出设备统称为外部设备。人们习惯上把主机和输入、输出设备这些看得见、摸得着的东西，叫做硬件，而把各种各样预先编制好的程序，称为软件。一套电子计算机系统，必须包括完备的硬件和丰富的软件。程序和指令要靠硬件去执行，没有硬件，软件就失去物质基础，无法实现；反过来，有了软件，硬件才能发挥作用，才有实际意义，否则也是一堆废物。

2-2 微处理器

运算器和控制器是微型计算机系统的核心，称为CPU，也叫做中央处理机，是由大规模集成电路构成，集成在一个芯片或几个芯片上，有时也称为微处理器 (Microprocessor)。微处理器只是微型计算机的一个组成部分，也是核心部分。

为了了解微型计算机的基本部件、基本概念和基本工作原理，让我们看一下图2·2·2所示的微型计算机结构图。从图中看出，由微处理器，内存贮器和输入输出设备的接口电路组成主机。主机通过接口电路再和输入、输出设备相连接。它们结合在一起，才是一个微型计算机系统。它们相互之间，通过三组总线，即地址总线、数据总线和控制总线来连接。

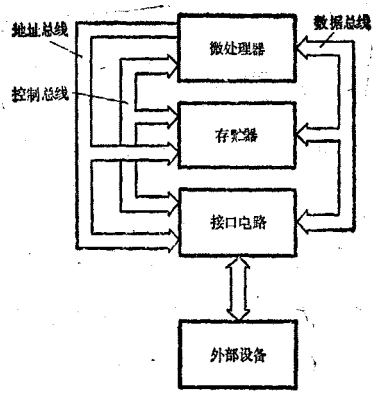


图 2·2·1 微型计算机结构图

把微处理器和键盘、显示器三者组合起来，便形成普及型计算机。第一章里所介绍的普及型计算机，把微处理器和键盘（输入设备）做在一个塑料外壳里，统称为主机，再用电视机当显示器，组成了最简单的电子计算机，有时也称为“个人计算机”

2-3 数据总线、控制总线和地址总线

如上所述，计算机五个部件相互之间，通过三组总线（控制总线、数据总线和地址总线）进行联系。控制器（微处理器中的一部分）发出的控制信号，经控制总线传到存贮器、运算器和输入、输出设备（通过接口电路），这些控制对象向控制器发回的信息也通过控制总线传输。

计算机处理的数据经数据总线由输入设备送到存贮器或运算器，而处理后所得的结果也经数据总线传到输出设备向外输出。

地址总线用于传送数据在存贮器中的“地址”。例如控制器要把某一数据从存贮器取出，就先要将该数据的“地址”经地址总线送到存贮器中。

输入/输出接口起什么作用呢？微处理器是工作在低电压、小信号条件下，且运行速度很快，但是它控制的对象，如显示器、打印机以及各种工业自动化设备，都是在高电压、大电流（相对而言）条件下工作，而且一般工作速度都比较低。输入/输出接口设备就在这二者之间起相互适配的作用。

2-4 字长和位数

微型计算机的运算器只能对二进制数进行运算。我们通常使用的计数制是十进制，有0～9十个数，逢十进一。但是计算机不能识别十进制数。

那么什么是二进制呢？具有两种不同状态的事物是屡见不鲜的。如灯只有亮和暗两种状态，晶体管工作时只有导通和截止两种状态，门电路只有开和关两种状态，卡片只有穿孔与不穿孔两种情况，等等。在计算机中，用符号“0”和“1”表示上述两种对立的状态。“0”和“1”叫做代码，它们不同于十进制数0～9中的数字0和1。“0”、“1”代码遵循“二进制”的运算法则。计算机能够识别“0”和“1”代码，能够接受由“0”和“1”组成的二进制数据。

大家都知道，在打电报时，都要经过编码及译码过程，例如要打一“大”字，首先要译成电码“1129”。这种电码叫“莫尔斯码”。每一个汉字都由四个阿拉伯数字组成。在电子计算机中的存贮器等也只能“认识”二进位的数码（即0与1），所以把几个数码组成一组来作为存、取或运算的单位。

这一组数码就好象电报中的电码一样，看作一个字。如果用四个数码组成一组——一个字，那么可以说，计算机的“字长”是4位。一般说，计算机愈复杂，它的字长就愈长，例如微型计算机常用4位、8位及16位字长，很少用32位的；而大型计算机则多是32位的。这是因为字的位数越多，它能容纳的信息量也越大的缘故。一台微型计算机的字长决定于其运算器中所能寄存的位数。这个位数又进一步决定了微处理器一次所能操作的位数。所以字长是表示运算器性能的主要技术指标。每种微处理器或微型计算机都有自己一定的字长。所谓4位机、8位机、16位机、32位机，指的就是其字长为4位、8位、16位、32位。例如，8位机中，运算器能寄存8位二进制数，数据总线能传送8位二进制数。

2-5 运算器和控制器

运算器里包含几个“寄存器”，用来存放直接进行运算的数据。当运算器工作时，进行某种运算之前，先要把数据从存贮器里取出来，暂存到运算器自身的“寄存器”里去，这叫做“取数”操作。有时又要把计算结果由运算器的寄存器存放到存贮器里去，这叫“存数”操作。计算机进行运算时，不停地大量从事取数和存数操作。

运算速度是衡量运算器质量高低的主要依据，而衡量运算速度的基本指标，叫时钟周期，有时也以时钟频率来表示。用时钟周期表示计算机工作节拍的长短，一般计算机做一次运算要几个时钟周期，至少也要用一个周期。用时钟频率则表示一秒钟内运算次数的多少。4位机的时钟频率一般为1MHz，8位机的时钟频率一般为2～4MHz，16位机的时钟频率一般为5

～10MHz,32位机的时钟频率一般要达到20MHz以上。

控制器是计算机的控制中心，它控制整个计算机有条不紊地工作，把计算过程及原始数据从存贮器中取出，并控制运算器实现自动运算。所以说，控制器好比是人脑的神经中枢，计算机所以能自动高速度地工作，都是在它的控制下实现的。

2-6 存贮器及其容量

存贮器是计算机中极为重要的组成部分，存贮器是存放原始数据、计算程序、运算中间结果和最终结果以及其它信息的记忆装置。有了它，计算机才能有记忆功能，才能把要计算和处理的数据以及程序存入计算机，使计算机脱离人们的直接干予，而在一定范围内自动地工作。

存贮器里存放的指令和数据越多，即存贮容量越大，计算机的“计算”能力就越强。存贮器的取数、存数、取指令的时间（或者叫“写入”时间和“读出”时间）越短，计算机的工作速度就越高。半导体存贮器的集成度越高，存贮器的体积就越小。半导体存贮器的容量是衡量集成电路集成度发展的标志。能够贮存1024个二进制数〔一位二进制数叫1比特（bit）〕的半导体存贮器集成电路，其容量称为1K位；能够贮存4096个二进制数字的存贮器，其容量为4K位，即每“1K”等于1024。所以，16K位就是16×1024=16384位，依此类推。七十年代末八十年代初已采用16K位存贮器，1981年研制出256K位存贮器，1982年出现512K位存贮器，1983年,100万位(1024K位）的超大规模集成电路也研制成功。1984年正研制1亿位存贮器，在一块绿豆大的芯片上，可以存贮每页400字的稿件三万二千张，计一千二百万字。

通常把8位二进制数（8比特），称为一个字节，字节的英文为Byte，简写为B。存贮器的容量通常就用KB和MB来表示。如果每个存贮单元存放一个字节，1KB的存贮容量等于存放1024个字节（即2^{10}个字节），叫做1K字节。同理，256KB就是具有256×1024个字节的存贮器容量，1MB＝1000K B就是具有一百万个字节的存贮器容量，100MB就是具有1亿个字节的存贮器容量。

前面说过计算机的运算速度是运算器的主要技术指标，不过运算速度不仅与运算器有关，而且还与存贮器的存取速度有关。对存贮器来说，容量和速度是相互矛盾的，容量越大的存贮器，写入和读出的速度越慢。为了达到速度快、容量大的要求，一般都把存贮器做成两类，一类容量较小而速度较快，另一类容量很大，但速度较慢。前者即内存贮器，或叫主存贮器。在微型机中，内存贮器都采用半导体集成电路。后者即外存贮器，如磁盘、磁带等外部贮存设备。内存贮器直接与运算器和控制器交换信息，参与操作；而外存贮器则存放大量数据和程序，定期成批地与内存贮器交换信息。这两种存贮器互相配合，互相补充。

2-7 只读存贮器(ROM)和随机存贮器(RAM)

内存贮器如按其功能分类，也可分为两类。一类叫只读存贮器，一类叫随机存取存贮器。只读存贮器存放固定不变的程序和数据，简写为ROM。ROM又有三种形式：①一般的ROM，其内容由厂家在制造过程中已做好，用户不能改变，只能读出它的内容；②PROM，用户自己可以根据自己的使用

要求，通过专门的写入装置，写进自己所编的程序，称为可编程序的只读存贮器，但只能编写一次；③EPROM，是一种可以多次改写的只读存贮器，写进去的内容可以擦掉，改写多次。EPROM写的速度较慢，而且需要一些额外条件，使用时仍作为只读存贮器来用。

随机存取存贮器简写为RAM。RAM的内容在计算机运算过程中既可以随时读出来，又可以随时写进去，所以用于存放随时要进行运算的数据，随时要使用的程序和其它要随时存、取的信息。

ROM电路比RAM简单，因而ROM的集成度比RAM高，成本低，而且当电源去掉以后，ROM中贮存的信息仍然保持，不会丢失。在计算机中尽可能地把一些管理、监控程序、汇编程序以及各种典型程序（如诊断程序、调试程序）放在ROM中。

半导体内存贮器从信息传送的方法来说，又可分为并行存贮器和串行存贮器。并行存贮器中字长的所有位同时进行存取，因此存取速度快，但设备较贵；串行存贮器则要一位一位存取，因此存取速度慢。目前微型机中都采用并行存贮器。

2-8 软磁盘和磁带

普及型微计算机的外存贮器都采用软磁盘和磁带。软磁盘是一圆形塑料薄膜片，形状和工作过程与塑料唱片差不多。它表面涂有磁性材料，以磁化轨迹（代替一般唱片的音槽）来记录信息。为了保护表面不受损伤，也和唱片一样，外面套有方形纸套。最常用的软磁盘直径为 $5\frac{1}{4}$ 英寸，也有3英寸和8英寸的。软盘表面用脉冲磁化成一圈一圈的磁道，每一圈磁道又分成若干

个区段，见图 2·8·1 。通常每个区段上存入 128 个字符的信息。例如，一种软盘有77圈磁道，每一磁道有26个区段，每个区段贮存 128 个字符信息，则一个软盘的贮存容量为77×26×128＝256256个字符信息。对“双密度”软盘，每个区段能贮存 256 个字符信息，总容量为上述容量的一倍。

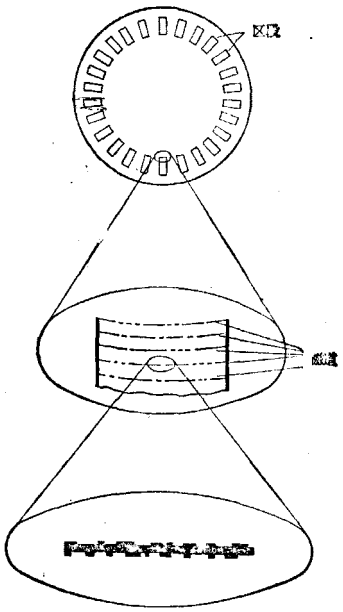


图 2·8·1 软盘上的磁道和区段

软盘可以根据磁道和区段的编号随时选取某一个磁道某一个区段上的信息，存取速度快，贮存容量大。

使用计算机的人可以收藏许多软磁盘，如同收藏许多唱片一样。可以买到预先全部录制好程序的软磁盘，也可以买到空白的软磁盘，用空白软磁盘录制你自己的程序，并且把这些录制的软磁盘保存在你的程序库中。

普及型计算机还大量使用磁带存贮器，既可以用一种计算机专用的盒式磁带，也可以用平常家庭使用的录音机的盒式磁带。磁带的贮存容量大，价格便宜。盒式磁带与软磁盘之间的主要区别在于：盒式磁带是一种顺序存取的装置，而软磁盘是一种可以随机存取的装置。在盒式磁带上，信息是以一连串的磁化点记录下来的，磁带上没有软磁盘上的区段和磁道所形成的统一的标准格式。在磁带上不能象在软磁盘上那样跳来跳去地读出，如果要读出盒式磁带上的第18个记录，就必须顺序经

过前面的17个记录。因此，要读出盒式磁带终点附近的信息，就需要费整个磁带很长的走带时间。此外，由于上述原因，在同一磁带上很难实现既读出又写入的操作。在使用普及型微计算机时，至少要准备两个盒式磁带部件，一个用于读出，另一个用于写入。

2-9 键盘、显示器和打印机

键盘、打印机和显示器作为计算机的外部设备，在本书第六章里要详细介绍它们的使用方法，这里只对它们在计算机系统中的作用作一概要说明。

键盘是向计算机输入信息用的一种外部设备。普及型计算机通过键盘把程序和数据逐个输入计算机，普及型微机还常把键盘和电路板组合在一起，装成一个整机。

除了键盘之外，卡片、穿孔纸带、磁盘、盒式磁带、传声器等等都可以作为信息输入设备。

输入信息除了字母、数字、符号、汉字外，也可以是图表、乐谱、电路、照片等。声音识别输入设备，则能把生产调度、交通指挥等场合的音频信号变成频谱图，再输入计算机。

打印机是计算机向外输出信息用的一种外部设备，能将输出的结果直接打印在纸上。和COMX35配套的打印机是一种热敏式打印机。它可以打印字母、图形、符号，还具有多种记忆操纵功能，并能直接观察记忆内容和直接更改记忆内容。

荧光屏显示器是人机之间进行交换信息的外部设备，可在屏幕上显示文字、数字、符号、图形。有的显示器还配有“光笔”，可以用来在显示屏幕上设计和修改电路、图形等。

电视机屏幕可用作计算机显示屏幕，但专用的计算机显示

屏幕和电视机屏幕的要求有所不同。电视机主要显示图象，所以黑白电视机除了黑白外，还有深浅不同的灰色级，彩色电视机要有不同的彩色和色度，但不需要分辩字符和图形的细节。计算机荧光屏显示器要求很高的分辩率，以便将尺寸很小而又书写很美观的字符或比例严格的图形准确显示出来，但对图象的灰度层次则无严格要求。

2-10 程序、语言和软件

上面介绍的组成计算机的各个部件，都属于计算机硬件。光有硬件，计算机还是干不成事，要让计算机真正解决各种问题，必须给它编制程序。为了运行、管理和维修计算机，就要编制出各种程序，这种程序的总和就称为软件。

语言是人们交流信息的重要工具，人与计算机交流信息也要使用"语言"。用二进制代码（0及1）编写出来的程序，就称为机器语言程序。机器语言的优点是能为计算机直接接受，缺点是人不易理解和记忆。

为了便于编写程序，人们用易于记忆的符号来表示容易混淆的数字代码，于是出现了符号语言。符号语言使程序、指令易于理解易于记忆，便于交流。但是，机器还是只认得机器码（即二进制数码），所以用符号语言编程序必须经过翻译，变成用二进制码表示的程序，机器才能识别和执行。开始，这种翻译工作是由程序编制人员用手工完成的，后来就编一个程序让机器来完成上述的翻译工作。具有这种功能的程序为汇编程序。汇编程序把符号语言程序翻译成机器语言程序，从这个意义上说，符号语言也叫做汇编语言。

汇编语言的语句与机器指令是一一对应的，程序的语句数

仍然很多，编程序仍然是一件十分繁杂、细致的工作，而且用汇编语言编写程序必须对机器的各种指令十分熟悉，一一对应。这就是说，用汇编语言编的程序只能用在特定的机型，还不能在不同的机型上通用。

为了使用户编程序更容易，为了让程序中所用的语句与实际问题更接近，而且使用户可以不必了解具体的机器，就能编写程序，经过人们不断的实践和创新，又出现了各种高级语言，叫做程序设计语言。高级语言是由表达各种意义的英语单词或短句加上数学公式，按照一定的语言规则构成的。BASIC语言是其中一种，普及型微机都用BASIC语言。高级语言直观、易于理解、学习和掌握，用高级语言编程序十分方便，大大减少了编程工作量。

显然，在计算机工作时，必须把高级语言程序翻译成机器语言才行。把高级语言翻译成机器语言的程序，称为编译程序或解释程序，所以高级语言也叫做编译语言。

本书介绍的普及型微机，还采用一种LOGO语言，因为BASIC语言没有绘图效能，LOGO语言是特为绘图而设的。LOGO语言简单易学，连小孩也会懂得用它来编出图形及音响效果。常用的还有一种叫FORTH的学术语言，专供教育及高速数学运算用。

将程序从一种语言转换成另一种语言，作为输入的语言叫源语言，相应的程序称为源程序；作为输出的语言叫目标语言或结果语言，相应的程序称为目标程序或结果程序。

操作系统和故障检查、诊断程序也是极为重要的软件。操作系统好像是乐队的指挥，和谐地指挥着每一个演奏者，使大家在指定的时间内完成指定的任务，按轻重缓急进行各种操作。当一台主机联有多台终端机时，操作系统使主机按一定时间

间隔轮流地分给各终端机用户使用，使每个用户都感到似乎他一个人在单独使用整台计算机似的。常用的操作系统有CP/M、CDOS（克罗门柯磁盘操作系统）、UNIX、RMX/186型等。

故障检查和诊断程序是这样一种软件，放在计算机里，一旦计算机发生故障，检查和诊断程序就开始运行，迅速准确地查出故障所在，发出报警信号，还能自动排除一般性故障。所以有了诊断程序，就不必用人工一一查找故障了。

此外，还有机器的监控管理程序、调试程序等。

为了扩大计算机的功能，便于用户使用，机器中还设置了各种标准子程序。这些子程序的总和，形成所谓子程序库。

上面所介绍的这些由机器的设计者提供，为了使用和管理计算机而编制的软件，统称为系统软件。

与系统软件相应的另一类软件是应用软件。什么是应用软件呢？用户利用计算机以及它所提供的各种系统软件，编制解决用户各种实际问题的程序，这类软件就称为应用软件。

人们常常说的“软件包”，是指一些应用软件的组合。由于应用软件逐步走向标准化、模块化，逐步形成了解决各种典型问题的应用程序的组合。

随着计算机硬件和软件的发展，计算机应用日益普及和发展。它们都要大量地处理数据，检查、建立各种表格。这些数据和表格按一定规律组织起来，使检索更迅速，处理更方便，也更便于用户使用，这就是所谓的数据库。

2-11 微型计算机结构

计算机的硬件与软件结合，组成一个完整的计算机系统，

大致包括：

一、主机

1.微处理器

（1）运算器：（a）算术运算部件；（b）寄存器。

（2）控制器

2.存贮器

（1）随机存贮器（RAM）

（2）只读存贮器（ROM）

（3）可编程只读存贮器（PROM）

（4）可擦除编程只读存贮器（EPROM）

3.系统总线

（1）数据总线

（2）地址总线

（3）控制总线

4.输入/输出电路（I/O接口）

（1）串行输入/输出接口

（2）并行输入/输出接口

5.时钟

二、电源

三、系统软件

1.汇编程序

2.编译程序

3.操作系统

4.控制程序

5.诊断程序

6.子程序库

四、输入、输出设备

1.控制打字机

（1）输出打印机

（2）电传打字机

2.显示器（CRT）

3.外存贮器

（1）磁带

（2）磁盘

（3）纸带

第三章　几种普及型计算机

3-1　普及型计算机系统

计算机虽有电脑之称，但是它完全是按照操作人员输送给它的命令和程序进行工作的。人们向计算机中输入命令和程序有许多种方法。普及型计算机输入信息的方便之处，是它采用了“人机对话”的方法。这就是按照BASIC语言的规定，有顺序地将各种命令、数据和程序，用键盘一个字符、一个字符地输入计算机。同时计算机可将这些命令、数据和程序全部在显示器上显示出来，以供你检查和核对。所以，普及型计算机除主机之外，必不可少的就是键盘和显示器。微型计算机主机的电路板体积是很小的，因此普及型计算机都将其主机电路板和键盘做在一起，装在一个塑料外壳中。键盘装在上表面，主机板在键盘的下面，成为一个整体。

实践证明，BASIC语言是一种方便的高级语言，用BASIC语言编写出的程序用途广泛。由于这个原因，普及型计算机（国外也常称“个人电脑”）多数使用BASIC语言。在它

的键盘上有BASIC语言中所用的字符按键和控制按键；计算机内部存贮器中装有BASIC语言的解释程序。这种解释程序是用计算机可以理解的语言——机器语言写成的，并且“固化”在普及型计算机的存贮单元中。靠解释程序的帮助，计算机就会识别从键盘送进来的各种字符和控制信号，而且还能正确地将它们组成一段段程序。计算机接到运行命令之后，编好的程序就会在计算机中运行，完成各种运算和数据处理。

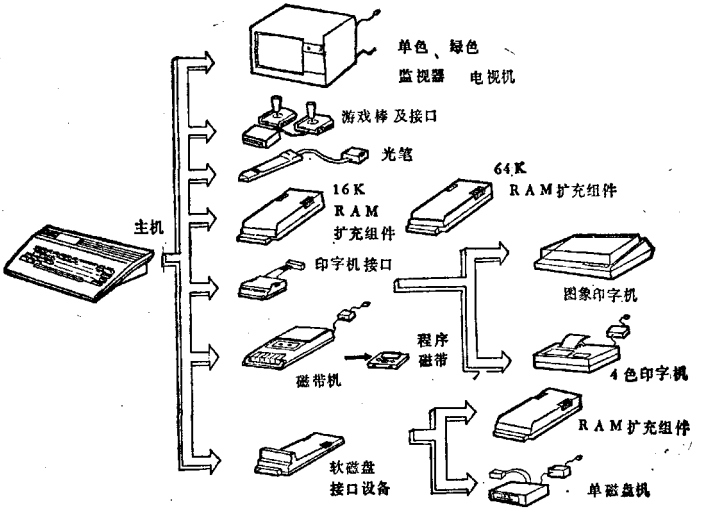


图 3·1·1 普及型计算机系统组成

至于BASIC语言的含义，将在本书第五章里系统介绍，当你认真学习好第五章之后，你就会懂得什么是BASIC语言，和如何运用BASIC语言了。

计算机主机与外部设备相连接时，将专用电缆的一个插头插入主机上的插口，另一头插入外部设备的相应插口。这种连

接在第一章里已作过介绍。对不同型号的计算机，还要请你参阅一下计算机的使用手册。因为插口多，连接电缆也多，容易混淆，注意不要连错。

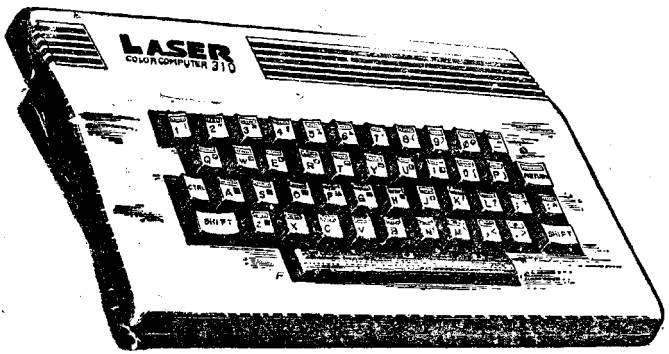


图 3·1·2　一种微型计算机的主机外形

普及型计算机的显示器一般可采用家用的黑白电视机，也可使用专用的绿色（或黑白的）屏幕的显示器（监视器）。大部分普及型计算机都具有彩色显示功能，如果有条件使用彩色电视机作为显示器，那就更好了。

除了主机键盘和显示器外，另一个不可少的部分是电源设备。普及型计算机的电源设备，一般单独做在一个塑料盒中。它有一根带插头的电源线，接交流220V电源；还有一根直流（电压一般为10V左右）输出线，带有一个专用的插头，插入主机插孔后，计算机就接上了电源。

由此看来，一个简单的普及型计算机也不是由一件东西组成，而是由带键盘的主机、显示器和电源三件配成“套”，才能进行工作。习惯上称这三部分组成一个普及型计算机的基本系统。

为了扩大普及型计算机的功能，设计制造者往往还为主机

配上许多其他外部设备，如印字机、盒式磁带录音机和游戏棒等。

磁带录音机用任何形式的都可以，但带速要稳，高频特性要好一些。在这里它不是用来录放歌曲和语言，而是录制二进制的编码信号，如10010111和11010100等。计算机工作时，它的存贮器中存有编好的BASIC语言程序，但是一旦关断电源，这些存贮的程序就会丢失。有了数据记录器（磁带录音机），就可以将这些程序录在录音磁带上。这时你可以放心地关闭计算机电源。当下次你再想使用这些程序时，可将这些存在磁带中的程序重新调入计算机的内存贮器中，继续使用。

既然数据记录器可以用磁带将程序、数据等信息长久地保存起来，计算机制造者和编制软件的人就将许多有用的程序存在一盘盘磁带上，作为软件产品提供给用户。这些编好的程序有数学计算程序，物理、化学等教学程序，各种游戏程序，检查计算　是否正常工作的检测程序，等等。

标准的 8 位或16位微型机系统中都有打印机，它将计算机中需要打印的图形、文字、程序、数据等内容打印在打印纸上，作为永久的记录。普及型计算机用的打印机，工作原理和一般计算机用的打印机略有不同。这里我们叫它“印字机”，以示区别。印字机体积比较小，打印出来的纸条也比较窄。印字机一般有四支颜色不同的笔，用来作图和印字，因此有时它又叫 4 色印字机。

有了印字机，你可以将一些较好的程序印出来。或者编制一些绘图程序，让计算机绘出千变万化或各种规则的图形。

游戏操纵杆（或叫游戏棒）是为了配合计算机游戏程序而设计的。在许多游戏程序中，显示器屏幕上显示出许多运动的图形，例如飞机、炮弹、汽车、皮球等，游戏操纵杆就是让使

用者操纵这些运动的物体来进行计算机电视游戏的，如让炮弹打中飞机、皮球飞进球门等。

下面介绍几种常见的普及型计算机。

3-2 R-1型微型计算机

R-1微型机是普及型计算机中的简易机，图3·2·1为其主机外形图。为了降低成本，它的键盘按键由软塑料制成。R-1的许多功能都设计得比较直接和简单，加上它有一些音乐和画图功能，学习起来比较方便，很适合小学生和初学者使用。

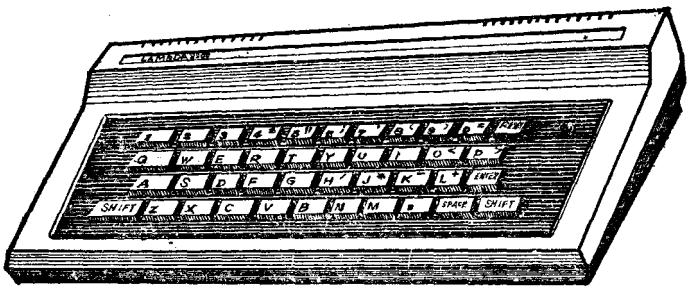


图 3·2·1 UR-1机主机外形图

R-1机采用Z-80微处理器作为主机的中央处理单元（CPU），主机配有8K字节的只读存贮器（ROM），内装BASIC语言解释程序。另配有2K字节的随机（读写）存贮器（RAM），供用户存放编写的程序和数据。

作为一个简易微型机，R-1主机和键盘做在一个小巧的硬塑料外壳中。R-1微型机共有42个键，为个适合青少年的学习和使用，每个键的字符都设计的简单明了。另外还安排了

一些固定图形按键和类似计算器固定功能的按键，例如×、÷、√、SIN、COS……所有这些设计都会给使用者带来兴趣和方便。

配在R-1微型机上的BASIC语言的语句和功能，能够完成编写程序的练习和运行很多实用的BASIC语言程序。和R-1机的整个特点一致，该机的BASIC语言比较简单、实用，相应的功能虽少些，但初学者学习却比较容易；它有作图和音乐功能，更能增加学习者的兴趣。使用R-1微型机应注意到它的特点以及和标准微型计算机的区别。

在R-1微型机的后面装有与外接设备相连的插孔。每一台R-1机配有一个交直流变换器，它将220V交流电转换为9V的直流电，并可通过电源插孔送入R-1机。R-1机中装有直流稳压电路，将9V直流电压变为+5V的稳定直流电压，供给全部微型机电路。另外还有显示器插孔和录音机插孔，可分别通过专用电缆接显示器（或电视机）和录音机。内存扩充（扩展）插孔插接内存扩充器（扩展器），可灵活地将R-1机的内存RAM从2K字节扩充到16K字节或32K字节。

3-3 LASER-310微型计算机

LASER-310型微型机属于普及型计算机中的中档机，它配有标准的BASIC语言，很适合初学者和中学生使用。专门学习BASIC语言的培训班，也可用它作为实习用机。

LASER-310的中央处理单元（CPU）选用Z-80A微处理器。它的时钟频率为3.54MHz。在主机的电路板上装有16K个只读存贮单元，也就是常说的16KROM。其中放有BASIC语言的解释程序。另外装有18K读写存贮器（RAM），用

来存放使用者编写的BASIC语言程序。

主机表面装有45个硬塑键盘，键盘式样和字符排列与标准8位微型计算机键盘相似。这种键盘使用起来比较方便，敲键速度也比较快。除几个特殊功能键之外，每个键上都印有两种字母、数字或符号（这三者统称为字符）。直接按某一键，显示器上就出现最明显的字符。按住 SHIFT 键再按该键，则启动该键的另一功能，显示器上显示该键上印有的另一字符。详细的用法及其特殊功能键的使用方法，我们在第一章有过介绍和举例。请根据所讲的方法和道理，并参照说明书，去熟悉每个按键的内容和功能。

LASER-310机可以使用专用的31厘米单色监示器，更好的是用彩色监示器或彩色电视机作为显示器。显示器通过随机配有的电缆线与主机相连。如果使用计算机专用监示器，应将电缆插头插入“VIDEO OUTPUT”插孔中；若用彩色电视机作显示器，则应把电缆插头插入主机的“OUTPUT”插孔。

LASER-310机在显示器屏幕上可以显示16行×32列的字符。如果显示图形，整个屏幕总共可显示128×64个点，这些小点子可以有8种颜色（如果用彩色显示器的话）。如果你要计算机显示9种颜色，则只能显示64×32个点。

除了主机和显示器之外，LASER-310机还配有许多外部设备，图3·1·1所示的系统就是LASER-310系统，图3·1·2是该系统主机外形。下面介绍各种外部设备的功能：

游戏操纵杆——玩电视游戏时使用。

光笔——在一定的软件程序管理下，通过光笔在显示屏表面上移动，屏幕上可显示相应的图形线条。

16K/64KRAM扩充组件——接上它可以将RAM存贮区

再扩大16K或64K，可以编写更长的程序在计算机上运行。

4 色印字机——我们看到，在它和主机之间有一个连接部件，称为“打印机（或印字机）接口”。许多复杂的外部设备和主机相连时，往往都需要加一个专用的“接口”设备。接口设备内部由许多电路组成，加上接插件，可以完成两个设备之间的适配，使它们协调工作。

数据记录器——即图中所示的磁带机。用于长久存放程序，以备需要时使用，可用一般磁带录音机代替。

软磁盘机——它的内部有一张类似软塑料唱片似的 5 英寸圆形软磁盘，其上涂有和录音带上一样的磁粉。它的作用和录音带一样，用来存贮计算机内部的程序等信息。这种软磁盘有很多优点：存贮容量很大，存入和取出的速度很快。磁盘机（也叫磁盘驱动器）就是使软磁盘转动并存、取信号的设备，其作用和电唱盘的角色相似。当然，“拾音器”用与录音机中磁头相似的组件。

LASER-310型机配有的BASIC语言，采用了美国标准MICROSOFT BASIC语言。这个版本的BASIC语言功能较强，语句完善，在世界上使用也很广泛，学习起来也比较容易，是一个比较好的程序设计语言。

3-4 COMX-35型微型计算机

COMX-35型微型机的功能与LASER-310型相似，但比后者要更强一些。它有比LASER-310型更大的内存容量，有32KRAM供用户使用。它的计算机集成电路选用美国1800系列电路。COMX-35计算机中央处理器为1802ACPU，有16K ROM和35KRAM（但使用者可用的RAM为32K字节）。它

的存贮容量在普及型微机中是比较大的，使用该机时一般不用再接扩充RAM的组件。16K的ROM中装有先进版本的BASIC语言，COMX BASIC是一个半编译程序，用途比较广泛。另外，COMX-35机还可工作于另一种计算机语言——FORTH语言。

显示器可以用彩色电视机或专用显示器。COMX-35显示文本模式为24行×40列，即整个屏幕写满可显示24行，每行40个字符（包括空格，空格也规定为一个字符）。COMX-35机可产生8种颜色，显示出的背景、字符及光标分别为不同的颜色。

COMX-35机内装有专用的小扬声器，可发出六十四度音程、十六级音量的各种不同音响。

COMX-35主机键盘有55个键。键上共有64个ASCII码字符（有的一个键上有两个ASCII码）。图1·1·1（a）就是COMX-35形主机图，从图中可以看到键盘的形状。

从图中可以看到COMX-35机键盘右边有一小杆，这就是游戏操纵杆。操纵杆除了供游戏使用外还代替了普通键盘的光标指示键。

COMX-35普及型计算机除了带键盘主机和显示器之外，也有较多的外部设备，与LASER-310机外部设备有很多相同之处，这里就不一一介绍了。

3-5 HKC8800微型计算机

HKC8800机实际上是一种标准的8位微型计算机，它可用于大、中学校教学，机关文件档案管理，以及工厂管理等许多方面。作为教学用机，它可称为普及型计算机中的高档机。

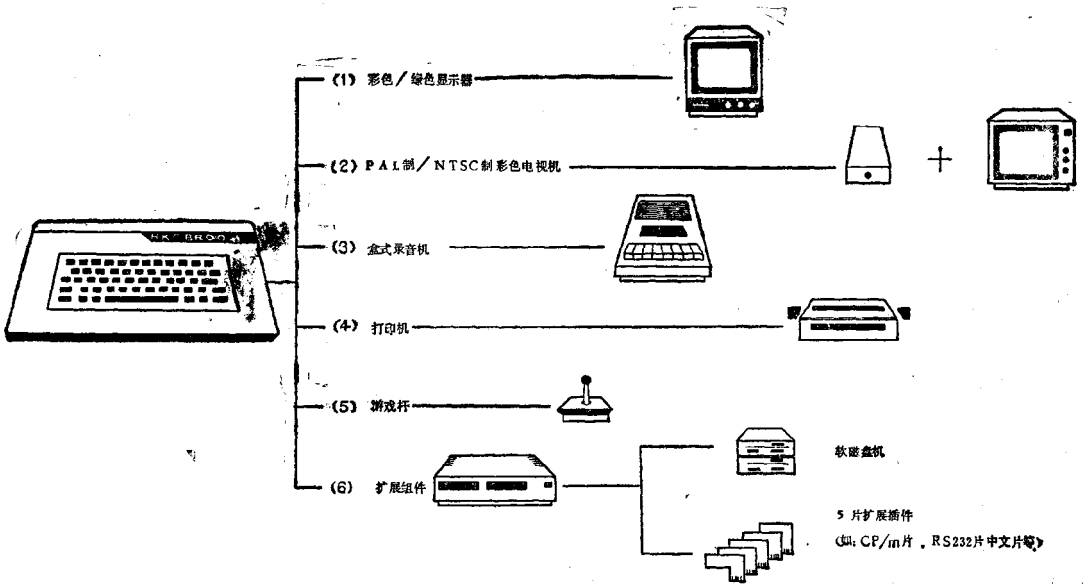


图 3·5·1 HKC8800微型计算机系统

HKC8800机设计的功能比较齐全，并且具有可扩展功能的设计，因而有广泛的用途。HKC8800机硬件上采用了先进的C-MOS集成电路，具有工作稳定、功耗低的特点。软件上它与流行的APPLE-II型机完全兼容，因而APPLE-II微机上丰富的软件可以在HKC8800机上运行。

HKC8800机的主机中使用了两个微处理器，一个是65S-CO2 CPU作为主微处理器，另一个8048单片微处理机作为键盘专用微处理器。HKC8800机的存贮器中有16K字节的只读存贮器(ROM)，存放着系统监督程序及BASIC语言解释程序；还有64K随机存取存贮器(RAM)，便于使用者随时存取各种程序。另外HKC8800机还配有两部软磁盘机作为外部存贮器。它们都装在一个叫做扩展组件的机箱内。该扩展组件箱内还装有开关电源和类似APPLE-II微型机的一组五个扩展插座。利用这五个扩展插座，就可将APPLE-II机目前使用的各种扩展功能的插件，如CP/M卡、RS232卡、汉字卡等等接入主机，而使HKC8800扩展为功能更强的微计算机系统。

HKC8800机的键盘设计为标准的硬键盘，键盘按键的排列基本为标准的ASCII码方式，共有55个键。其中包括10个游标控制键和3个屏幕编辑控制键。HKC8800主机电路板及各种接口都与键盘装在一个扁平塑料外壳里，使用方便。

HKC8800机的显示器有35厘米和31厘米两种，为彩色/绿色监视器。若使用家庭彩色电视机作为监视器，应外接一个TVMODULE部件。监视器的屏幕上有三种显示模式：

1.文本模式(TEXTMODE)

可显示40列×20行英文文字字幕，每个英文字符均可在软件控制下显示正常（黑底白字）、反白（白底黑字）或者闪烁三种形式，在这种模式下不显示彩色。

2.图象模式

高分解象度模式——整个屏幕可显示280×192个图象点，每个点可改变6种颜色。

低分解象度模式——整个屏幕可显示40×48个图象点，每个点可改变16种颜色。

3.混合模式

上述的文本模式及图象模式，可以在屏幕上同时显现。在这种混合模式下屏幕的最下端可显示4行文字。

在HKC8800机的主机里，还装有三种接口电路：盒式录音机接口、打印机接口和游戏机接口。这样，HKC8800机可以很方便地连上录音机作为简便的外存贮器，连上打印机作为硬拷贝手段——这在标准8位机系统中是不可少的。另外HKC8800机配有一些游戏软件作为学校教育的辅助教材。通过游戏机接口接上游戏操纵杆，就可以方便地使用游戏程序了。

3-6 APPLE-II PLUS微型计算机系统

APPLE-II PLUS微型计算机系统（以下简称APPLE-II机系统）是一种比较典型的8位微型计算机系统。它结构灵活、功能较全，而且使用简便。世界上许多公司为这种系统研制了非常丰富的软件和扩充组件，使得它的应用领域越来越广泛。这种计算机成为当前最流行的个人计算机之一。

作为一个计算机系统，它的硬、软件配置比较齐全。从硬件设备来看，这种系统的基本配置包括主机和扩充组件两大部分。

一、主机

主机采用6502微处理器作为CPU，其主频为1.023MHz。

它具有12K字节的只读存贮器(ROM)和48K字节的随机存贮器(RAM)。整个主机电路做在一块大电路板上，装入一个扁平的机壳内。主机电路板上还有监视器接口电路，音频盒式录音机接口电路、游戏操纵杆接口电路、扬声器接口电路及扬声器键盘接口电路及键盘等。这样，只要把监视器接到主机后右端的插座上作为标准输出设备，把录音机接到主机背后的插座上作为外部存贮设备，用主机本身所带的键盘作为标准输入设备，这套系统就可以工作了。电源开关装在主机背后左端。

键盘上共有52个键，包括二十六个英文字母键、十个数字键（0～9）以及八个符号键（=或－、←、→、＊或:、＋或；、<或，、>或．、?或/），一个空格键，两个转换键(SHIFT)，四个特殊键（ESC、CTRL、RESET、REPT）和一个回车键(RETURN)。

二、扩充组件

主机电路板的后部装有八个50线插座，可以插上任何扩充组件。这就可以在基本系统配置的基础上扩大功能，以满足不同的需要。

经常用到的扩充组件有：

（1）软磁盘驱动器及其接口电路。一个接口电路可带两台软盘驱动器，通常是用$5\frac{1}{4}$英寸的软磁盘驱动器。每片软磁盘存贮容量为140K字节。

（2）打印机接口电路板。用打印机作为输出装置可以产生出供长期保存的材料。

（3）存贮扩充电路板。可使RAM由48K扩充到64K。

（4）系统软件。APPLE机在主机的ROM中存有固化

软件，其中包括有一种扩展BASIC语言（称为APPLESOFT）的，解释程序。系统的自启动程序使用户工作起来很方便——一开机就可以使用扩展BASIC语言。当系统中配置有软磁盘驱动器时，在磁盘操作系统DOS的支持下，BASIC语言具有较强的磁盘文件处理能力——你的程序或数据都可以作为一个文件存入软磁盘，按文件名字对它们进行统一管理，大大地方便了用户。

以上我们对一些计算机做了简单介绍。如果你对其中某些地方不大懂，没有关系，可以跳过去接着读下去。这对你使用微型机影响不大，待看完本书全书，初步学会使用后，再回头来看，就可以看懂了。

3-7 普及型计算机的一般用途

前面介绍的几种普及型计算机，都由8位微处理器芯片组成，它们都属于“8位机”。目前国内外微型计算机中，8位机的应用最为普遍。

下面介绍8位普及型计算机的一般用途。

一、教育方面

普及型微型计算机是教师和学生的好伙伴。无论小学还是中学，各个年级、各门课程都可以编制出计算机软件，供教师和学生共同使用。例如，把历年的高考题、中学入学考试题汇编成计算机软件，使用者可以根据需要调用同类型的试题。如要物理学中的静电学题目，即可把历届的有关静电学考题一个一个调用，也可任意选调其中一部分。如果做不出来，计算机会给你提示；如果做错了，计算机会给你指正。最后还会告诉

你标准答案。又如，按小学各个年级的数学教学内容编成的软件，可以用作课上学习指导和课下自学练习。使用这种软件，计算机屏幕上会自动显示各种数学公式和习题，由浅入深，从易到难，逐步深入。当你理解困难的时候，计算机会给出相应的提示，启发你从哪里入手思考，诱导你得出正确的回答。计算机还能帮助你检验掌握某些知识的程度，帮助你补课，帮助你质疑，最后还会根据你的回答评出成绩，并告诉你得了多少分，够得上个什么水平。

普及型计算机可帮助你学会使用计算机，学会用BASIC语言编写程序，培养你的逻辑思维能力，以及综合运用知识的能力。你可以从掌握普及型计算机入门，逐步学习计算机硬件知识和软件知识；从计算机在教学中的应用着手，逐步扩大到应用在图书馆、学生俱乐部、教务工作等不同场合。

计算机不仅能帮助你学习数学、物理、化学等知识，还可以帮助你学习音乐。你可以运用BASIC语言的音乐语句，编出你所喜爱的乐曲来。编成之后，可以当即检验，让计算机奏出你所作的乐曲。如果你不满意，还可以借助显示器屏幕来修改乐谱。

无论在高中、初中甚至小学，都会遇到各种各样的作图题，计算机能帮助你作图和制表。例如用计算机制表，可以在一到二个小时之内，完成你往常需要花几天时间才能完成的工作。完备的计算机系统还配有打印机或印字机，它会将你编好的程序和图表印在纸上。所以计算机既能帮助你设计图表，还能帮助你填写内容，统计数字，免去你的许多烦琐工作。

二、办公自动化和事务管理方面

计算机将成为办公人员和管理人员的可靠助手。它可以将报告、文件、人事档案、科技档案材料分门别类地贮存起来；

在你需要的时候，随时调出，显示在屏幕上，供你查阅和使用。如果你想抄写一份，打印机会在几分钟内给你印出整个内容。利用计算机的分析判断功能，可以非常方便地对文件材料进行分类、检索、查找、修改、增减、统计等各种处理，而且会在很短时间内高效地完成这些工作。

计算机可以作为一名优秀的“会计”，利用它高速度、准确无误的运算能力，完成现金、成本、存贷管理、营业帐目等方面的核算和统计。原先一个会计师很长时间才能算完的帐目，如果用计算机，顷刻之间就可完成，而且分毫不差。如果你将财务金融方面的资料输送给计算机，它还可以进行市场分析和预测、资金流动情况的预测等等，充当一个经验丰富的“会计师”角色。

三、家庭和个人应用方面

普及型计算机在很多国家已进入家庭，成了人们的好管家。计算机可以帮助人们计算电费、水费、煤气费、房费，并进行统计和记录。计算机可以根据家庭的收入和生活需要作出开支预算。用计算机贮存亲友、同事的姓名、地址、电话号码等资料，还具有检索、查找方便的优点。在国外，人们还用计算机管理和计算日常的各种税务、存款借款的利息，进行股票及债券分析，等等。总之，凡是涉及计算、统计、存贮等方面的工作都可交给普及型计算机去完成。

普及型计算机和显示器配上游戏程序软件（这些游戏程序你自己也可以设计）和游戏专用操纵杆，就会成为一个高级电子玩具游戏机。计算机在游戏中能与你“斗智斗勇”，比速度，比反应，变幻无穷。人们在紧张的工作和学习之余，用计算机做一会儿有趣的游戏，将是一件十分惬意的事。

第四章　程序设计与流程图

我们已经知道，电子计算机是一种能在一定范围内自动处理信息的机器。它能按照使用者的具体要求去自动地完成一定的任务。所谓“自动”，其实就是按照使用者的一系列命令一步一步地执行。这种为了完成一定的任务而由人们事先计划好的一系列命令，就叫程序。

4-1　指令与程序

一、指令

指令就是一道命令，它指示计算机去完成某一具体操作步骤。

一条指令一般来说应该包括两部分内容：操作类型和操作对象。比如：让机器把3和5相加，这是一个指令。其中，“加”是操作类型，3和5是操作对象。

前面讲到过，计算机的特点之一是有“存贮”能力。它的内存贮器分为数以千计的小单元。犹如一座有很多房间的大楼中，每个房间都有一个房间号码一样，存贮器中的每个单元也

有一个地址码。比如第132号单元（通常地址码用二进制数表示）。一条指令中涉及的操作对象和操作结果通常都放在存贮器的某个单元中，所以在指令中都用这些单元相应的地址码来代表这些数。用二进制数表示是由于计算机只能识别二进制数，实际上操作类型也是用二进制数表示的称为操作码。因此，一条指令是由操作码和地址码组成的。象上面举到的那个例子，如果事先已经把3和5分别放到名字为“A”和为“B”的两个单元中，并准备把计算结果（8）放到名字为“C”的单元中，那么这就要用三条指令才能完成。“从A取出3并放于累加器”、“从B取出5也放入累加器”以及“把结果8存入C”各为一条指令。要注意的是，“从B取出5放入累加器”，这一条指令也同时执行了“把3和5相加”的任务。

二、程序

为解决某一问题而设计的一系列指令就组成一个程序。换句话说，程序也就是解决某一问题的一套步骤。

计算机执行一条指令所做的事情虽然有限，但按照一连串指令一步步地做下去，就可以完成一项复杂的作业任务。把一个复杂问题的解决过程，分解为一步一步简单的操作步骤，再相应地变成一条条指令，这就是程序设计。

程序似乎是个新概念。其实，我们每个人在日常生活、工作中做许多事情都是在执行一定的程序。就拿吃一顿饺子来说吧，要经过和面、备馅、包饺子、煮饺子这样几大步骤。细致些说，备馅又分为买、洗、切、拌等若干步骤。只不过这个“程序”已成为生活经验，“存贮”在我们自己的脑子中，没有明显地表示出来罢了。如果你想要做个新式样的菜就不大有把握了，于是去查菜谱。其实菜谱不就是用文字书写的做一道

菜的“程序”吗？日常生活是如此，用计算机解决某个问题也是如此。

比如，要计算一下全班同学某门课程的平均成绩，这项工作的程序用文字叙述起来就应当是：

1.逐个记录每个同学这门功课的成绩；

2.累计分数；

3.累计参加测验的人数；

4.求平均值；

5.输出平均成绩。

再比如，对全班同学某门课程的成绩分类归并，求出成绩为：0～9、10～19、20～29、……90～99、100分的学生人数。程序可以是：

1.准备一把豆子和十一个碗并分别标上0、1、2……9、10号；

2.询问一个学生的成绩；

3.看这个成绩的十位数字是几，就在几号碗里放上一粒豆子；

4.检查全班同学是否都问过一遍了，如果没有，转到第2步；

5.如果全班都问过一遍了，就统计各个碗中的豆粒数并输出，这就是所要的结果。

总之，设计一个正确的程序是十分重要的。计算机自动工作的过程，实质上是执行程序的过程。机器再好，终归是个机器，没有指令和程序，什么也干不出来。

4-2 流程图

一、流程

程序一方面说明了各个操作步骤，另一方面也说明了这些步骤执行的先后顺序，也就是作业流程。这些内容除了用文字叙述（象上面说的那样）以外，更直观的是用流程图来描述。流程图又叫程序框图，它用一些几何图形（框和线）附加简短的文字符号把操作步骤与顺序流向清晰地描述出来。比如上面讲过的那个做饺子的例子，其过程用流程图表示出来就是图4·2·1的样子。在流程图中，用矩形框表示一般的处理功能，叫处理框；用带箭头的线表示流程的路线和方向，叫流线。画流程图，可以帮助我们准确无误地构思程序，在比较复杂的问题中确定各操作步骤之间的相互关系。

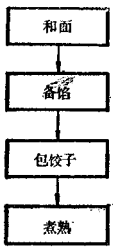


图 4·2·1 包饺子的流程

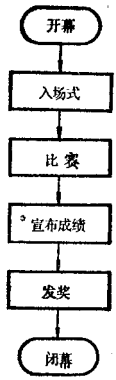


图 4·2·2 运动会流程

二、典型的流程

各种工作的处理过程尽管有千差万别，但只要我们稍加分析就可以看出，经常用到的不外是这样几种情况：

（1）顺序执行的流程

比如学校开运动会，运动会的整个过程可以用图 4·2·2 所

示的流程图表示出来。这里六大步骤必须按一定的次序顺序执行，不能颠倒过来。图中首尾两个框具有弧形边线，这种框叫起止框，用来表示一个流程的开始或结束。

（2）具有分支的流程

我们以运动会上跳高比赛为例。一个运动员参加跳高比赛的过程是这样的：当跳高比赛开始点名时，运动员应去报到，然后参加预赛，检查试跳成绩，成绩合格则可以参加决赛，否则只好退出比赛。这里不仅仅是简单地顺序执行各步骤了。检查试跳成绩之后的下一步骤不能事先确定，有两个可能，应视“检查试跳成绩”的结果而定。遇到这种有两个可能的去向时需要经过判断才能确定的情况，在流程图上就会出现“分支”。上述过程的流程图如图4·2·3所示。在流程图中，用一个菱形框表示“分支”，它叫判断框。判断框有一个入口和两个出口。判断框的里面写上判断检查的内容，也就是询问某种条件是否得到满足，如果满足了则走一个出口；如果不满足，则走另一个出口。应当弄清楚，尽管在图上一个菱形框有两个出口，但一个运动员按上述流程前进时却只能走其中一条路，经过菱形框后只能有一个去向，不可能既参加决赛又退出比赛。

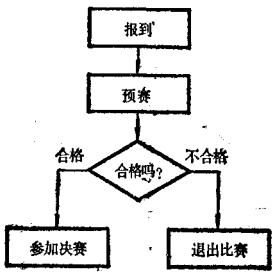


图 4·2·3 跳高比赛流程（1）

（3）具有跳跃的流程：

如果运动会的比赛规则规定，凡是上届运动会获得跳高前三名的运动员，可以不必参加预赛而直接进入决赛，流程图就应作相应的变化，如图4·2·4所示。这里，一个运动员报到后，首先要经过一次检查，判断他是否属于上届比赛的前三名。如

果不属于，那么后面的流程就与图4·2·3类似了。如果属于上届的前三名之内，则可以跨越“参加预赛”和“检查试跳成绩（合格吗？）”这两步而直接用流线表示进入“参加决赛”那一步。从图中可以看出，“参加决赛”这个处理框有两个入口，表明参加决赛的运动员有两个来源。当然，对于一个运动员来说，只能属于这两个来源中的一个。

（4）具有循环的流程

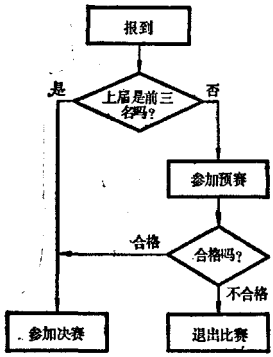


图 4·2·4 跳高比赛流程（2）

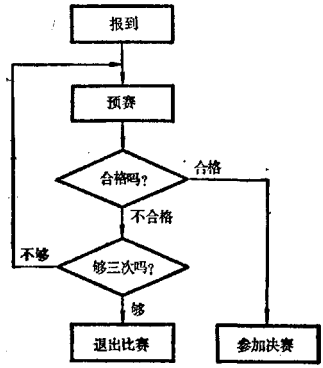


图 4·2·5 跳高比赛流程（3）

如果比赛规则规定，运动员参加跳高预赛可以试跳三次，只有三次成绩均不合格者才被淘汰，否则就准予参加决赛，流程图该怎样体现出这一规则呢？我们来看看图4·2·5。当检查试跳成绩后，如果成绩不合格，就进入第二项检查——判断这个运动员试跳的次数是否已经达到三次。如果已经达到三次，只好退出比赛；如果还没有达到三次，就用流线表示一种返回跳跃——跳回到“预赛”那个处理框去。“预赛”这个处理框就有了两个入口。从这个流程图我们可以看出：“预赛”处理框和两个判断框之间的流线形成了一个闭合的环，这就叫循

环。顺着这个环走一遍，就是执行了一次循环。在我们这个例子中，这个循环最多被执行三次。有一条流线从环外进入（由“报到”框进来），这叫这个循环的入口；有两条流线离开这个环指向环外的“参加决赛”和“退出比赛”两个框，叫这个循环的出口。在这个例子中，循环有两个出口，表明一个运动员或者由于在某次试跳中成绩合格而退出循环去参加决赛；或者由于三次试跳都不合格而退出循环、被淘汰。

4-3 用流程图帮助设计计算机程序

我们已经明瞭了什么叫流程图和几种流程的例子。把这些简单而典型的情况按不同需要组合起来，就可以表达一些较复杂的过程。在设计一个计算机程序时，流程图是个很有用的工具。在筹划一个计算处理过程时，人们可以边构思边画流程图，把一个比较复杂的问题分解为一系列比较简单的步骤，并把各个步骤之间的关系和执行顺序，简单明瞭地勾画出来。下面我们通过一些简单的例子，帮助读者逐渐熟悉这种构思的过程。

例1 假设全班共有45名同学参加某门课程的测验，其中得5分的有11名，得4分的有18名，得3分的有14名，得2分的2名。要求计算出全班的平均成绩。

这项工作的处理程序可以用简单的顺序流程图表达出来（见图4·3·1）。在本例中，我们要用到五个存贮单元来存放运算的对象和结果。为了方便，我们给这五个单元各取一个名字，分别叫A、B、C、D、E。首先，我们把得5分的人数（11）送入名为“A”的存贮单元里。这种把某一数值送入一个存贮单元的动作在计算机的工作里是经常会遇到的，我们称

之为“赋给”。把11赋给A，记为11→A，意思就是把11这个数值送入地址名字为A的单元中存放起来。这里，顺便多说一句：类似地，如果我们写A→W则表示把名字叫A的那个单元里存放着的那个数送给名字叫W的存贮单元。也就是使W单元也存有与A单元中同样的数。

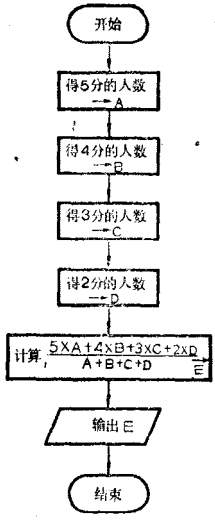


图 4·3·1 计算平均成绩

本例中再往下的几步都和第一步相似，分别地把得4分、3分、2分的人数（即18、14和2）赋给B、C、D单元。然后就该计算平均分了。流程图的这一步骤在框内标记为$\frac{5\times A+4\times B+3\times C+2\times D}{A+B+C+D}\rightarrow E$，意思就是：把A、B、C、D四个单元中存放的数值分别乘上5、4、3、2再相加起来（其实就是总分数和）作为被除数，把A、B、C、D四个单元中的数相加（就是总人数）作为除数，求得的商（也就是平均分的值）再送入名为E的单元中存放。至此，总平均分已经计算出来，但这项工作还不能算完。因为上述计算是在机器内进行的，使用机器的人并没有看见，所以还应该有一步：让计算机把计算结果（它现在已经存在E单元中了）输送出来，比如说是在电视屏幕上显示出来，或印字机上印出来。这种处理操作在流程图中用平行四边形框表示，叫输入输出框。在这个框里面，写上“输出E”，表示把E单元内所含的数值输送出来，显示给使用者看。

例 2　假如首先把 3 和 5 这两个数分别赋给 A 单元和 B 单元，然后再把它们的位置交换一下，即 A 单元和 B 单元中的内容互相交换一下。可能有人会说：这很简单，只要两步操作就够了，把 A 单元的内容赋给 B 单元，把 B 单元的内容赋给 A 即可。请你想想看，是这样吗？

把一个数值送入一个存贮单元和从一个存贮单元中取(读)出一个数值这两种操作，与在录音磁带上录一支歌曲和听一支歌曲的情况很类似，向一个单元里送入一个新内容就会把原来的旧内容冲掉而代之以新内容；从一个单元里读出一个内容则该内容在这个单元内仍然保留着，并不因此而被破坏掉。有人把这种情况称做“取不尽，挤即走”。

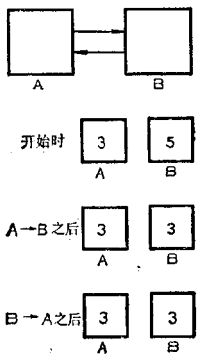


图 4·3·2　交换两个存贮单元的内容

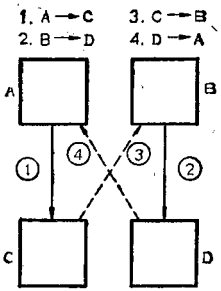


图 4·3·3　交换两个存贮单元的内容

图4·3·2很清楚地告诉我们，用“A→B”和“B→A”两步操作达不到上面提出的“交换”目的。因为当把A单元的内容赋给B单元以后，B单元的内容（5）就被“3”挤掉，也变成为“3”了。这时再进行“B单元内容赋给A单元”（B→A）操作是没有意义的。

那么怎么办呢？我们来看看图4·3·3。这里再用两个存贮单元（分别命名为“C”和“D”）。按照图中所示意的步骤进行“赋给”操作，就可以顺利地完成“交换”任务。C、D两个单元的作用是在交换过程中充当临时存放的场所。那么有没有更好的办法呢？请再看图4·3·4。这里只要有一个单元充当临时存放场所就够了，不过各单元中互相“赋给”的操作顺序一定要按图4·3·4中所示意的那样进行，不可颠倒。这时整个程序的流程图就如图4·3·5所示。

例3　在铁路车站托运行李，如果到某地的运费规定为：行李重量不超过50公斤，每公斤收运费0.15元，超过50公斤，超过的部分每公斤运费多收0.10元。请编制一个根据货物重量计算运费的流程图。

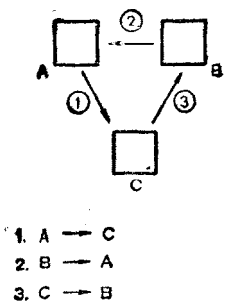


图 4·3·4　交换两个存贮单元的内容

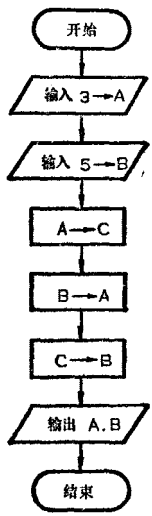


图 4·3·5　交换两个存贮单元的内容（流程图）

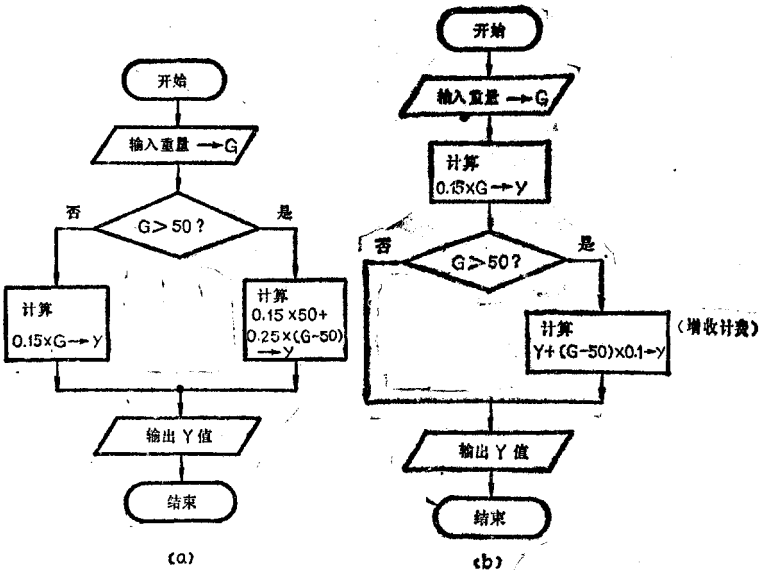


图 4·3·6 托运费用计算流程图

在这个问题里，由于行李的重量不同时应当采用不同的计费标准，显然应当采用带有分支形式的流程。在流程图中安排一个分支时，要特别注意在何处分开，在何处汇合。如图4·3·6（a），采用的是一种并列式的分支，当判断重量G是否大于50公斤之后，根据“是”或“非”两种判断结果分别采用两种公式计算运费，采用其中任意一种公式计算之后下一步都是“打印运费值”。也可以象图4·3·6（b）那样采用跨越式的分支。无论重量是多少，都先按每公斤0.15元计算，然后再判断重量是否大于50公斤。如果大于50公斤，则把原计算的运费增加一部分（对超过50公斤的重量，每公斤增收0.10元）；如果重量不大于50公斤，则跨过“增收计费”这一步（即维持原计算的运费不变），直接到“打印运费”（输出Y值）这一步

了。

在图4·3·6（b）中，“增收计费”的处理框中采取了这样一种写法：Y＋(G－50)×0.1⟶Y。它的含义是把Y单元中原来存放的数值取出来，加上增收的运费部分〔(G－50)×0.1〕，然后又存入Y单元中。这种Y＋（某个数值）→Y的处理方式是计算机中常用的一种操作方式。这实际是一种“累加”运算。注意，在这种情况下，操作前后在Y单元中存放的数值不一样了。

例4 连续把一百个数送进计算机，求这些数的总和。

当然，可以象本节例1中的做法那样，在机器内部用一百个存贮单元分别存上这一百个数，然后再让机器把这一百个单元的内容相加起来。但是这种做法是一种笨办法，因为它占用的存贮单元太多了。如果问题仅仅要求我们把总和求出来，而不必把这一百个数都记忆住，就可以采用前面例3中所用过的“累加”的方法。除了用一个存贮单元A存放每次送进来的一个数之外，另用一个存贮单元（例如为W）作为累加器。每次送进来的数都再加到这个累加器里。下一次又送进一个数来，仍存入A单元（注意：这时A单元中原存的上次送进的数就被“挤掉”了）。然后再把它加到W单元中去……如此做下去，可以想见，A单元中总是存放着刚送进来的那个数，而W单元中存放着已送进来的所有数的“和”。所以当一百个数全部送完以后，W单元中存的就是这一百个数的“总和”了。

现在我们来看看这个总流程图（图4·3·7）。这里有两个问题要注意。首先，作为累加器的W单元，就好象一个存钱的储蓄罐子一样，显然在一开始时应当是空的，也就是说累加器的初始值应当为0。所以在流程图的一开始有一步是“把0赋给W单元”，意思是先清一下库，准备“进新货”。其次，我

们看到，每输入一个数后都要做同样的动作，即：把新输入的数赋给A单元，再把它累加到W单元中去。这一套动作要重复进行一百次。很自然地我们会想到，用循环式的流程来实现它最合适，当然希望是有控制的有条件的循环——循环了预定次数之后就停下来，退出循环——而不要成为那种无休止的循环。如何实现对循环的控制呢？有几种方法。我们在这个例子中先介绍其中一种。在这里我们已经知道要输入一百个数，换句话说，循环次数已经预定为一百次，所以可以直接用“次数”来控制。当“次数”到100以后就退出循环。那么又如何知道“次数”呢？我们可以再利用一个存贮单元作为“计数器”（例如把它名为I单元）。用它来累计循环的次数：开始时I单元内的值赋为1（这样第一个数输入时I的值为1），以后每循环一次则加1。而在每次循环的末尾，都判断一下：I的值是不是大于100（实际循环次数是99次）？如果不大于就继续循环——跳回到“输入一个数”那一步去重复执行——否则就退出循环。流程图4·3·7就是结合考虑这两个问题后画出来的。

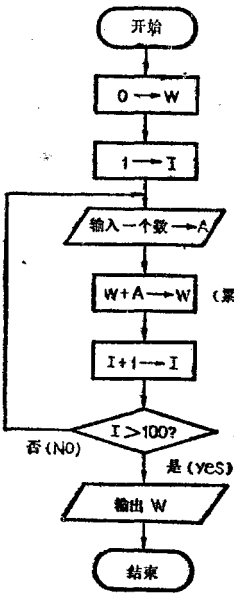


图 4·3·7 累加流程图

例5 连续送入若干个数，它们的值在10～100范围之内，求这些数的连乘积。

完成这项工作的思路与例4相类似，不过前者是“累加”，

这里则是“连乘”。同样要注意上面提到的那两个问题。第一个问题是：W单元在这里作为“连乘器”，即每一个新送入的数都要与W单元中原来的数相乘，乘积仍存入W单元中去，于是W单元存放的数等于已输入的各个数的连乘积。既然如此，W单元的初始值就不应再是0而应当是1。第二个问题是：在本例中，由于事先并不知道输入数字的确切个数，所以不能用检查循环次数的办法来控制循环。那末，怎样才能不至于无休止地循环下去呢？我们注意到：输入的数有个特点——都在10～100之间。利用这个特点，当所有数都输入之后（即我们希望不再循环下去的时候），再人为地多送进一个数。这个数明显地远离上述指定的范围，比方说是个负数。于是，流程中，在每送进一个数之后增加一项检查，检查它是否小于0？如果小于0了，说明它是我们人为增加的那个数，也就是表明“所有数都送完了”的信号。于是流程转为退出循环。这就是我们要介绍的第二种控制循环的办法——用一个所谓“终止标记”（见图4·3·8）。图中同时用I单元累计输入数个数。

例6　我国人民四化建设的宏伟目标是：在本世纪的最后廿年内实现工农业年总产值翻两番（以1980年为基准）。请编一个程序，计算一下年平均增长率需要达到百分之几才能实现这一目标？

设1980年的总产值为1，翻两番就意味着到2000年应达到4或4以上。如果年增长率为1％，则1981年总产值就是1.01，1982年总产值就是$1.01\times1.01=1.01^2$……2000年总产值就是1.01^{20}。如果年增长率为2％，20年后总产值就是1.02^{20}……循此思路我们就可以画出流程图了。这里也要用到循环的流程：让年增长率依次取百分之一、二、三……每次循环都计算一下2000年的产值（存入A单元）。如何控制循环呢？显然，

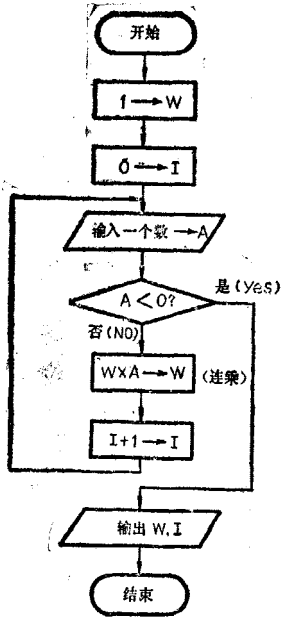


图 4·3·8 连乘流程图

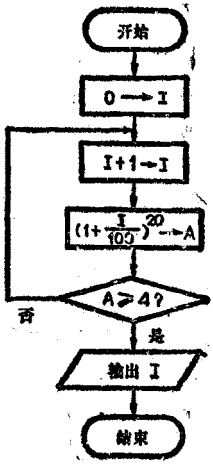


图 4·3·9 计算平均增长率的流程图

当2000年总产值一旦大于或等于4就该退出循环了。这就是又一种控制方法——用计算结果来控制（见图4·3·9）。

例7 任意送进三个数，要求计算机把它们从大到小顺序打印出来。

设输入的三个数分别存放在地址名字为A、B、C的三个存贮单元中。当然不能就按A、B、C的顺序输出打印。如何实现按大小顺序输出，可以有几种不同方法，读者可以自己试试。这里介绍的一种流程图（图4·3·10），当然不见得就是最好的方案。它的思路是：先比较A和B的大小，如果A＜B，就把A、B两个单元中的值交换（交换的方法见例2）。这

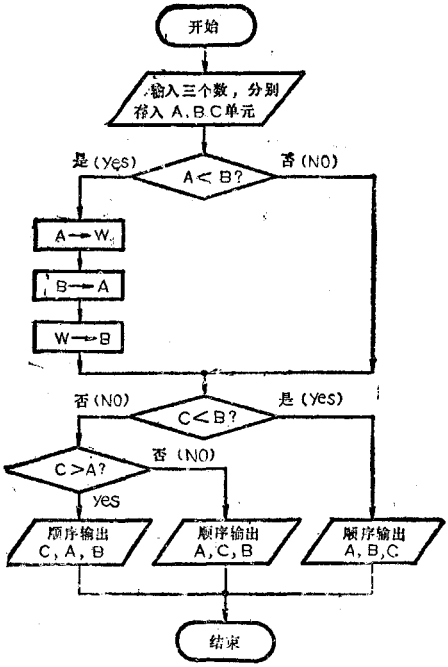


图 4·3·10 三个数按大小顺序排列的流程图

样，在流程图上，无论经由哪一条路径到达图中“C<B?”框时，A单元的值都不小于B单元的值。然后再根据C单元的值是否比B单元的还小（或是比A单元的还大）决定总的输出顺序。

例8 把一个班40名同学某课程的成绩（百分制）都输入计算机，把最高分与最低分挑出来。

流程见图4·3·11。这里也采用循环，每次送入一个分数参加比较。当所有分数都送完以后再送入一个大于200的数作为“终止标记”以便退出循环。

这里采用了2个存贮单元（名为H和L），第一次输入的

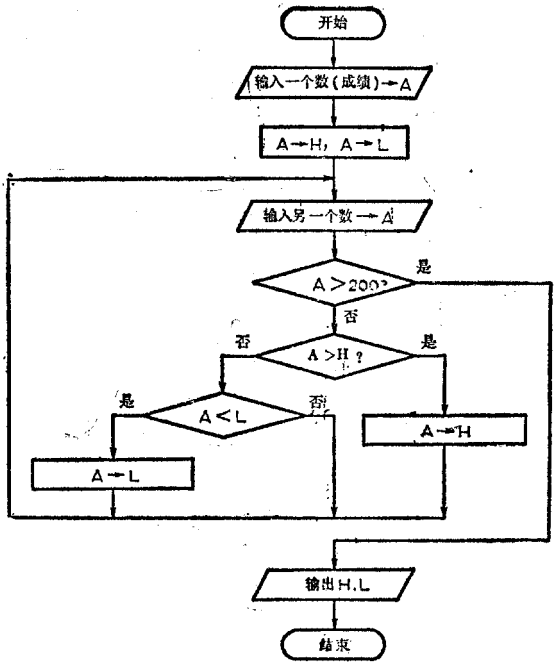


图 4·3·11 挑出最高最低分的流程图

分数先分别赋给这两个单元。以后每次送进来的分数都与它们相比较。如果它比H单元存的分数还高（或比L单元的还低）则把这个分数送入H（或L）单元顶替其原来的分数。这样H和L单元里将总是存着到当时为止的最高和最低分数值。

例9 要求把这40名同学的成绩按从高到低的顺序排列起来并输出。*

*注 作为一个综合性的练习，我们举这样一个稍为复杂一些的例子。初学的读者如果感到有困难，可以跨过这个例子。这对阅读后面的章节不会有障碍。待将来对于编写程序比较熟悉以后再回过头来看本例就不会感到困难了。

为了排序，需要对这40个分数多次进行两两比较。所以，必须把它们放进40个存贮单元中去，而且这40个单元应当有一个顺序编号。为此，我们给这40个单元起一个集体的名字“A”，且用A（1）作为其中第一个单元的名字、用A（2）作为其中第二个单元的名字……用A（40）作为其中第40个单元的名字。然后开始排序，其思路是：（参见图4·3·12）先以第一个成绩为基准，用第二个与它比，若第二个高，则两个的数值交换，否则不交换。再用第三个与第一个比，第四个与第一个比……直到第四十个与第一个相比，看谁分数高，以决定是否交换。经过这样一轮比较、交换之后，放在第一个单元里的肯定是最高的分数值。然后，再以第二个单元里的值为准，依次用第三、四……四十单元的值与之比较和做必要的交换。这一轮下来，放在第二个单元里的肯定就是居第二位高的分数值。如此一轮一轮地做下去，就可以把全部40个单元里的分数值从高到低按顺序排列好了。图4·3·12的流程就是根据上面的思路编制的。

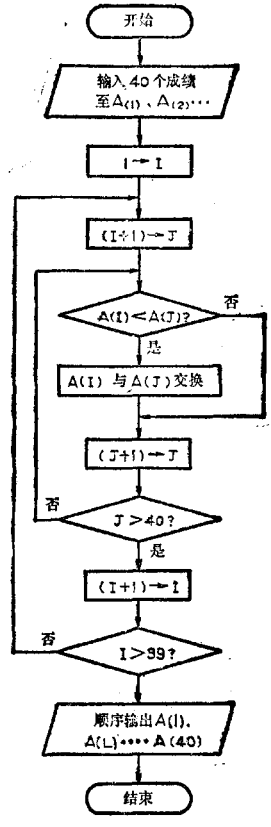


图 4·3·12 40个数排序的流程

以上，通过详细讨论九个例子，说明了如何用流程图帮助我们设计程序，也附带地介绍了一些简单的常用编程技巧。读

者可以想象到，更复杂一些的问题，也完全可以经过类似的分析，综合应用各种简单流程的办法描述出来。通常，对一个较复杂的问题，可以采用先粗后细的方法，就是说先把整个问题分为几大步骤，再把大步骤细分为较小的步骤……相应地，先画出粗略的流程图，再把其中每一框细分为较详细一些的流程图，同时不断改进。

设计出的程序，首先要正确，即能走得通，而且能达到预期目的。此外，还要考虑它的质量高低。怎样评价一个程序的质量，什么样的程序才算是一个好程序，不是一个简单的问题。初学者可以先不必拘泥于此。我们在这里仅简单地提出三点注意，即：一是程序要简单；二是交给计算机去执行时运行速度高，省时间；三要求尽量少占用存贮单元。为了实现这些要求，需要从各个角度周密地安排，比如在流程上既要条理清楚，又要尽量减少不必要的重复或“回头路”、“弯路”，存贮单元尽量重复使用等等，这里面还有不少技巧。归纳起来，一是时间，二是空间，一个好的程序，应该既快又省。其实，在一定程度上也可以说，在所有计算机软件设计中，时间与空间是一对矛盾统一体，需要根据实际情况权衡解决。

学习完这一章，你已经初步掌握了用流程图来描述你所构思出来的程序。在下一章你将学习怎样使用BASIC语言把程序写给计算机去“看”。因为这章所讲的所有流程，计算机是看不懂的。你在学习下一章时，可以边学习BASIC语言的各种“语句”，边对照着本章讲到的流程图，并试着把本章中九个例子的程序用BASIC语言写出来。

第五章　用BASIC语言来设计程序

计算机的各种功能是靠计算机执行一定的程序来实现的。编写计算机程序，可以使用多种语言，在普及型计算机中，主要用BASIC语言。BASIC语言的全称是“初学者通用符号指令代码”语言。BASIC五个字母就是由全称中五个英文单词的字首组成的。BASIC语言简单易学，使用方便，是一种入门语言。

用来编写程序的BASIC语言，是一些命令和语句，计算机能够识别它们，并按照这些命令、语句去执行。

5-1　BASIC语言的常用语句

大家知道，儿童学说话，是从学说简单的单词、短句开始，进而学习复杂的语言的。学习BASIC语言也是这样，要先学习一些简单命令和语句，编一些简单的程序，然后逐步学习全部的语句和用法，掌握编较复杂程序的方法。最重要的是还得上计算机操作和经常编写程序，调试程序。只有熟悉了BASIC语言的语句和用法，有一定的编写程序的经验技巧，才能自如

地编写出水平较高的程序。

在BASIC语言中有些类似英文单词的“命令”，如NEW、LIST、RUN等等。它们命令计算机立即去执行某类固定的工作。前面谈到过，计算机一接通电源，进入BASIC语言工作状态，显示屏上就会出现◆符号(有些型号计算机是]或>—等符号)，表示计算机已经处于等待命令的状态。这时你可以键入某个命令，再按下回车键，计算机就会立即响应，执行并完成你输入的命令。

BASIC语言的另一部分“英文单词”被称之为语句。如LET、PRINT、GOTO、END…。这些语句加上运算符号和数字，再按照顺序排列起来，就可以组成一段“真正”的程序。例如计算一个半径为3 cm的圆的面积，计算公式为$S=3.14159\times3^2$($S=\pi R^2$)，可以用语句将这个运算写成一个短小程序：

```
10    LET    A = 3 * 3
20    LET    S = 3.14159 * A
30    PRINT  S
40    END
```

这四行每一行为一个语句，每一行最左边的数字如10、20、30等称为语句标号，它们将程序语句按标号从小到大排列起来。可见标号起到了将不同的语句区分开来和规定语句的执行顺序（由小到大）的作用。同时，这些语句标号告诉计算机，将语句贮存起来，而不是立即执行。

你一定会注意到每行语句之前的语句标号并不是按自然数1、2、3、4…依次排列，而是10、20、30、40…这样使用的。这样做的好处是必要时可以在两行语句之间，加入一行或几行新的语句。这在编写和修改程序时很有用。因为我们编写

的程序往往不会一次就完好无误，常会遇到要在编好的语句之间再补充一些语句的情况。在计算机BASIC语言中规定，有效的行数值是从0到65529的自然数，而且计算机执行语句时是从小到大执行。例如在第10句与第20句之间若没有其它语句，则执行完第10句之后就执行第20句。不将语句标号一个数紧接一个数，就避免了在具体操作时，增添新语句时的困难。另外，请注意每行结束后还要按下回车键，否则在显示屏上将出现不分行的程序。

从上例中可以看到BASIC语言程序中使用了一些数学计算符号，这些符号和平时的数学运算符号大致一样。有个别的不同，那是为了计算机显示方便而规定的：

-	负号
↑	乘方号
*	乘号
/	除号
+	加号
-	减号

为了让你能尽快地在计算机上编写程序，这一节先介绍几个命令和简单的语句。先请看NEW、LIST、RUN这三个命令。

NEW 这是“清除”命令，用来清除计算机原有程序，或者叫“清零”。

计算机开机后，在你使用之前可能有人输入过一些程序，或者无意中按动了某一按键。总之，在计算机的内存贮器中可能已存入了一些与你的程序无关的内容。因此在你编写新的程序之前必须先键入“NEW”命令，将原存的东西清除掉。做法是顺序按下N、E、W键然后按回车键：

NEW CR

这时计算机就会自动清除原有内容。如果要使用计算机编程序而又不准备保留以前的什么内容时，建议你每次编程序之前先用NEW命令，这样可以保证你所编写的新程序不会与旧的内容相混。

这里再说明一下回车键。它又称为换行键或执行键。一般是在每一行内容输入完之后按下该键，表示该行的内容输入完毕。在不同型号的普及型计算机中，该键的标记不同，用得较多的是RETURN，还有ENTER、RTN等等，其功能却是一样的。在BASIC语言中，语句或命令有长有短，你用键盘逐个字符地输入它们，何时输入完毕，就全靠按回车键来表示。为了简便，在以后的例子中规定用CR来表示回车键。由于每句之后都要用，一般写程序时就略去不写出来了。

LIST

这是“开清单”（或叫“列表”）命令。它命令计算机在屏幕上显示出内存贮器中的全部程序。按照下面讲的方法试几次，你就会知道LIST命令的意义。

如前所述，用NEW命令可以清除计算机中所有内容，但是清除之前如果想看看内存贮器中究竟写入了什么内容的话，你可以使用LIST命令。方法是用键先输入L、I、S、T，然后按回车键。

LIST CR

计算机接到这个命令后，就会自动地将写在内存中的全部程序

从头到尾地列出来，显示在荧光屏上。这时你可以看到列出的内容中是否有需要保留的部分。

又如，当你刚刚写入一段新程序之后，用LIST 命 令，可以检查你输入的内容是否正确，从而决定增加和删改。

通常在 LIST CR 之后，计算机会将它内部存有的所有程序从头到尾列出，显示在屏幕上。如果你只需要看看写入程序的某一部分，比如你刚刚写入的第50到100句，可以输入 如 下命令：

LIST　　50,　　100 CR

这时屏幕上就只显示从50句到100句之间的语句。在这个 命 令中，语句标号50和100之间的逗号不能省略。

RUN

这是“运行”命令。它命令计算机执行存放在内存贮器中的程序。如果你 以 RUN CR 方式送入命令，计算机就从标号最小的程序语句开始运行。如果你送入 RUN50 CR 形 式 的 命令，计算机就不运行50句以前的语 句，而是从第50句 开 始 运行。BASIC 语言规定，跟在RUN之后的语句标号是任意的，程序从哪一条语句开始运行，就在RUN之后写上哪一 个 语 句的标号。

以上介绍了NEW、LIST、RUN三个命令，下面再 来 学习几个最简单、常用的BASIC语句。语句是程序 的 主 体，每段程序都是由一条一条的语句组成的。弄懂并记住了这几个语句，加上上面已学过的三个命令，你就可以进行一些简单的编程练习和计算了。

LET

为“赋值”语句。它的一般使用形式如下面几例所示：

```
10    LET    A=10
20    LET    B=25
30    LET    C=3+5*7-6/3
40    LET    I=0
70    LET    D=A+B-C-3
```

每个LET语句的意义是：将等号右边的值（数值或算式运算结果之值）赋给等号左边的字母变量。这与一般语言中“让”字的意义相似。例如第一语句就有“让A等于10”的意思。

计算机可用来进行数值计算，为了有效、简便地进行运算，BASIC语言中引入了一些英文字母（A、B、C…X、Y、Z），把它们作为变量来代替数字，就象初中代数中的字母变量一样。在一段程序中，变量的数值可以变化，但有一点必须注意，BASIC语言程序中的变量在进行运算之前，必须先将确定的数值赋给它们，也就是要用LET语句“赋值”。LET语句中的等号“=”与数学中的等号不完全相同，它的作用是将其右边的值赋给左边的变量，使左边变量的值为右边的值，请看下例：

```
10    LET    A=10
20    LET    A=A+3*5
30    PRINT  A
RUN
25
```

我们依次键入10到30这一段程序中的每一语句，再输入RUN命令，计算机即依次执行各条语句。当计算机执行完10句之后A的值即为10，再执行20句，A的值就变为25（10+3×5=

25）这个数值了。注意，这里等号右边的“A”其值还是原来的10，但等号左边的A，其值就不是10了，是重新赋值后的25。第30句的PRINT是打印语句，运行完第30句之后，计算机的显示屏上就会出现变量A的值25。

LET语句在BASIC语言程序中使用很多，几乎每一个程序都不止一次地要用到它。所以在BASIC语言中规定，LET语句中的LET三个字母可以省略不写，这样上面的例子就可写为：

```
10    A=10
20    A=A+3*5
30    PRINT
RUN
25
```

结果是一样的，使用中却简便多了。再强调一遍，上例中的10到30是程序，RUN是运行命令，25则是显示的结果。

PRINT

为“打印”语句。它的功能是将这一语句中的内容打印出来。执行这条语句，要打印的内容就会在显示屏上显示出来。PRINT语句可以编写在程序中，也可以用立即执行的方式进行。从下面的例子中可以看到它的特点。

```
10    A=3
20    B=4
30    C=A+B
40    PRINT    C
50    END
```

第50句END表示这段程序结束。输入了这段程序之后，再键入RUN命令，屏幕上就会在下一行显示出打印结果值7，即

变量C的结果。

如果用立即执行的方式运行PRINT语句，上例就可以这样完成：

```
PRINT    3 + 4
```

同样可以在显示屏上得出打印结果7。请注意，PRINT语句的前面，如果加上语句标号，就可作为语句写在程序中；如不加语句标号，就成为立即执行方式。

PRINT语句除了可以打印出变量的数值或者算式运算结果之外，还可以将字符原样打印出来，但需要打印的内容必须加上引号，请看下面两个例子。

例1 键入

```
PRINT "ABC"
```

时，屏幕上显示

```
ABC
```

例2 键入

```
PRINT "C=", 3 + 4
```

屏幕上显示

```
C=  7
```

在PRINT语句中，引号中的字母、符号原样照印，而不加引号的部分，则只打印出它运算结果的数值。例2中PRINT语句里有一个分号“;”，它是必须有的，它表明要打印的是两个内容，即“C=”和3 + 4的结果“7”（不是“3 + 4”）

BASIC语言中的引号不分左右，即没有“‘”和“’”的区别，只有一个“"”符号，它既可作引号的左半部分，又可作引号的右半部分。因而在计算机的键盘上，只安排一个按键就可以了。

在PRINT语句中，经常要用到分号“；”和逗号“，”。如果在一个PRINT语句中，需要打印几个内容，例如上面的例2中，要打印“C＝”和3＋4的结果“7”两项内容。这样一个PRINT语句中的几项之间，要用逗号“，”或者分号“；”分开。逗号与分号的不同点是：若使用分号，则打印出的两项之间连在一起；若使用逗号，则打印出的两项之间不连在一起，各自从某个固定位置开头。

例　键入

```
PRINT "A"; "B"; 35
```

屏幕显示

```
AB35
```

而若键入

```
PRINT "A", "B", 35
```

则显示

```
A          B
35
```

这些规定请你用手中的计算机试几次，了解它之后，会帮助你设计出满意的打印格式

PRINT语句在BASIC语言中也是一种常用而重要的语句，它将计算机运算的结果打印或显示出来。这样你就不会感觉计算机深不可测，而是可以通过PRINT语句，看到变量的数值和运算的结果。PRINT语句对编写、调试程序，以及得到程序的结果都是不可缺少的。

END

这是结束语句。在一段程序的最后必须加上这条语句。因为计算机在接到运行命令——RUN之后，会自动地按照语句标号的顺序由小到大地执行。如果一段程序不加END语句，

计算机就会一直不停地执行下去，执行完一段程序，又接着执行另一段程序，达不到你只想执行一段程序的目的。而且会把两段无关的程序相互混淆，造成错误。

使用END语句可以避免上述现象。END 语句一般写在一段完整程序的最末后。当程序运行到END语句时就会停止。你这时可检查上面一段程序的结果，并且判断分析所编写的程序是否正确。

在这一节中，简单介绍了NEW、LIST、RUN 三个命令和LET、PRINT、END三个语句，以及+、－、＊、/、↑这几个运算符号，你可以使用它们编写简单的程序。例如上一章例 1 中要计算全班45名同学的平均成绩，就可以写出如下的程序：

```
10    LET    A=11
20    LET    B=18
30    LET    C=14
40    LET    D=2
50    LET    E=(5*A+4*B+3*C+2
                 *D)/(A+B+C+D)
60    PRINT  E
70    END
```

将这段程序正确地输入到计算机中，再输入RUN 命令，显示器上就会出现正确的结果——全班同学的平均成绩。

5-2 数字、变量和表达式

一、数字

BASIC语言中使用的数字就是常用的阿拉伯数字，有0～9

十个数字。为了使用方便，采用十进制计数方式。数的表示与日常的记数方法相同，如“1826.37”、“－2643”等等。也可以使用科学记数法，例如＋0.283425E 9（即0.283425×10^{9}）和－0.729546E－3（即-0.729546×10^{-3}）等。其中数字部分称为尾数，一般均为六位有效数字，英文字母E后面的数称为指数（或阶码）。在数字之前的“＋”号，有时可以省略不写。这种科学记数法表示的数也称浮点数。计算机既能处理整数，也能处理浮点数。处理的整数是从－2147483647到＋2147483647，在这个范围内的整数都是精确的。浮点数的范围是从-0.170141×10^{39}到$+0.170141\times10^{39}$，尾数中的小数部分精确到6位数字。

一般普及型计算机有效的数值范围是$-10^{38}\leqslant x\leqslant10^{38}$。初学编程所进行的计算，一般都不会超出这个范围。

二、变量

计算机中主要进行的是数值计算。但是用计算机算题时，除了直接用数字运算外，BASIC语言还规定，和代数运算一样，可用英文字母作为变量，来参加运算。如你在前面已见到过的A＝3、B＝4、C＝B＝3中的A、B、C，就是三个变量。引入变量给计算机编程和运算带来了很大的方便。在BASIC语言中有三种变量，下面分别作一介绍。

1.简单变量——前面举例用过的A、B、C等都是简单变量。它由26个大写英文字母中的任意一个来表示。它还可由一个大写英文字母后跟一个0～9之间的阿拉伯数字组成，如A3、B4、X5、Y7…，这样加在一起，简单变量一共可有$26+26\times10=286$个。根据规定，两个英文字母，或数字在前、字母在后，例如AB、DE、4B、C27等形式的字母、数字组

合不能作为简单变量。

2.数组变量——也称带下标的变量，它和简单变量的作用类似。数组变量的形式一般为A（1）、B（4）、A（2，1）、D（2，2）等。

数组变量也可以象简单变量一样代表数值，它一般成组应用，如A（1）、A（2）、…A（8）、A（9）。这些被代表的数值往往具有某些共同的特性，为了有效地进行运算，BASIC语言规定了数组变量。数组变量用26个英文字母A～Z之一作为数组名字，用0～9十个数字写在字母后面的括号中作为下标。

3.字符串变量

在BASIC语言中，可以用A、C9等形式的变量，来表示某一个数值如53、27等。同样地，在BASIC语言中还规定了象A$、B$这样一类的变量，用来表示一串英文字母如ABCD、WANG等。如同在“A=31”之后A与31等同一样，在赋值语句A$=“WANG”之后，A$与WANG也是等同的。WANG称为字符串，A$称为字符串变量。关于字符串变量的规定和使用方法，在后面还要做专门的介绍。

三、表达式

表达式就如同一个数学式子，例如A+B-3*6/2-C。它是用算术运算符号加、减、乘、除、乘方及括号等把数字、变量以及函数等连接起来的式子。字符串也有表达式，它是将字符串常数或字符串变量用字符串连接符“+”连接起来的式子。在字符串两端要加引号。

表达式中形如3+5*7-9或B*B-4*A*C的叫做算术表达式，而形如“ABC”+“DEF”，A$+“1234”则

是字符串表达式。

5-3 运算符号和运算规则

运算符号是将数字和字母变量连接起来进行运算和操作的符号。每一个运算符号告诉计算机执行一个特定的操作。计算机的运算功能有很多种，相应的运算符号也比较多，大致分类有以下四类：

一、算术运算符号

这类符号执行与数学中数字运算相同的运算，符号也尽量采用数学中习惯使用的符号。下面按照运算的优先顺序列出这些运算符号：

算术运算符号	意义
（ ）	括号 第一优先级
↑（有的用∧）	乘方号 第二优先级
*	乘号 第三优先级
/	除号 第三优先级
+	加号 第四优先级
-	减号 第四优先级

这里有几点说明需要注意一下：

1.括号没有大、中、小括号之分，即没有大括号｛ ｝和中括号〔 〕，如果需要在括号中套括号，只需在（ ）内再套（ ）即可，如 3 + 5 * （9 *（4 - 2））。但是要注意必须将每一对括号的“（”和“）”写全。有时偶尔将右边“）”丢掉，计算机遇到这种情况就会给出错误信息而停止运算。例如上例写成 3 + 5 * （9 *（4 - 2）就是错误的。

2.乘方号“↑”，也称为自乘或指数。因为计算机显示中不便于在字母或数字的右上角显示数字、字母，所以乘方形式一律用“↑”表示（有的计算机中用∧表示）。这样3^2应写成3↑2，A^3应写成A↑3，B^C应写成B↑C。当然，若方次较少，把3^2写成3＊3，4^3写成4＊4＊4，A^2写成A＊A也是可以的。

3.乘、除号规定为“＊”和“／”，它们具有相同的运算优先等级。在一个算式中，如果同时有“＊”号和“／”号，则应按从左到右的顺序先后执行。使用除号时不能将“／”号左右的数字或变量分成上下，如$3/5$＊6不能写成3/5＊6。

4.加号和减号也具有相同的运算优先等级，在同一算式中也是按先左后右顺序执行。

二、关系运算符号

计算机的一个很大的优点，是它具有判断和处理的功能。BASIC语言中规定了如下的关系运算符号：

=	等于
< >	不等于
>	大于
<	小于
>=	大于或等于
<=	小于或等于

请注意，其中的不等号< >、大于或等于号>=、小于或等于号<=与我们平常所用的有些不同，使用中应按规定书写和输入，计算机键盘上是找不到≠、≥和≤符号的。另外，也找不着“< >”键，及“<=”或“>=”键，要顺序按下两个键（“<”及“>”；“<”及“=”；“>”及“=”）。

三、逻辑运算符号

使用BASIC语言还可以进行逻辑运算，它们的符号是：

AND	“与”运算
O R	“或”运算
NOT	“非”运算

这些运算符号帮助你进行逻辑运算，使用时应注意逻辑运算是在整数方式下使用。在做指定运算之前，逻辑算符将表达式转换成整数。另外，在NOT算符后面的表达式必须放在括号内。逻辑算符在运算中与“+”和“-”属同一优先级。

四、赋值算符

赋值算符实际上就是等号“=”，它一般多用于赋值语句之中，它的意义是“取什么值”，而不仅仅是“等于”。例如：

```
LET  B= 3
LET  A= 5
LET  B=B+A* 2
```

最后一句的意义是：计算B+A* 2的值并将结果赋给B。可见变量B的值在这句运行后会由3变为13。

五、其它运算符号

:	分隔同一行中的语句(用于在用一行中写几个语句)
;	PRINT的定界符
,	PRINT的定界符
(	组限界
)	组限界

\…\　八位二进制值

#　八位十六进制值

&　十六位十六进制值

以上介绍了BASIC语言中的运算符号，它们可以将数字和变量连接起来组成表达式，而计算机在进行计算时还要正确处理上述运算符号的操作顺序。这在BASIC语言中是有明确规定的。归纳起来说，在一个表达式中有许多运算符号时，运算次序是：

1.先进行括号内的运算，然后进行括号外的运算。如果有几层括号套在一起，即括号之中又有括号时，先计算最内层的括号，然后依次计算外一层的括号，直至最外层括号。

2.乘方“↑”。它是按由左向右的次序进行的。例如2↑3↑2相当于$(2^3)^2$，而不是2^9。计算机将先算出2的3次方8，然后再计算8的平方，得出64。

3.乘和除具有相同的运算优先次序，在表达式中从左到右，先遇到的先计算。加法和减法也是这样。例如3＋6＊5－15/5相当于$3+6\times5-\frac{15}{5}$。计算机先进行乘法6＊5，再进行除法15/5，然后做3＋30，最后运算33－3得出结果。

BASIC语言中的运算是严格的，如果忽视了它的运算规则，错用了运算符号，都不会得出正确的结果。

5-4　提供数据的语句

在用BASIC语言设计的程序中有许多变量如A、B、D2、X5等，这些变量在参加运算之前必须要赋给它一定的数值，否则计算机就认为该变量的值为零，请看下例：

```
10  PRINT  A+B
20  END
RUN
 0
```

可见，由于事先没有给变量A和B赋值，计算机将A和B的值都看作零对待了（或者给出一个极限的数值）。因此在BASIC语言中，象LET这样的给变量提供数值的语句显然是必不可少的。离了这种语句，计算机运算就无法进行。

普及型微计算机中提供数据的语句有三种：LET语句、INPUT语句和READ－DATA语句。

LET

LET语句是BASIC语言中最基本的一种提供数据的语句。前面已经初步介绍了它的规定和应用。在一段程序中，可能会对一个变量多次赋值，这也就要多次用到LET语句，整个的计算过程往往要对变量进行多次“赋值——计算——赋值”的处理。这一点你会在经常地读、写程序中体会到的。

INPUT

这条语句称为“键盘输入”语句。LET语句规定了在写程序时，必须将所要提供的数据写在程序中，例如：

```
10  LET  A = 3
20  LET  B = 5
30  LET  C = 3
40  LET  D = B*B - 4 *A*C
 ⋮
```

这就存在一个问题：如果我们想改变A、B、C的数据时，就必须重新改写程序，并再次输入到计算机中去：

```
10  LET  A = 4
```

```
20  LET  B = 9
30  LET  C = 5
40  LET  D = B * B - 4 * A * C
      ⋮
```

但是如我们有了INPUT语句之后，就会克服这种缺点。INPUT语句规定在该语句中只先设定变量而暂不提供数据。数据是等到程序开始运行之后，当计算机执行到INPUT语句时，由计算机在显示屏上打出问号“？”，表示计算机等待你为INPUT语句中的变量提供数据。你看到“？”之后，就可以用键盘将要赋给的数值输入，这时数据就会自动赋给相应的变量。请看下例：

```
10  INPUT  A
20  LET    A = A * A + 3
30  PRINT "A = "; A
40  END
    RUN
```

这时显示器屏幕上将出现“？”。如果你在键盘上按下“2”键，那么屏幕上即出现“A = 7”（因为第20语句中是 A = 2 × 2 + 3 = 7）。如果要第二次运算，那么你可再按一下“RUN”键，则屏幕上将再次出现“？”。当你输入一个新的数值“4”时，屏幕上将变成“A = 19”（因为 A = 4 × 4 + 3 = 19）。

BASIC语言还规定在INPUT之后可以带多个变量，例如执行

```
10  INPUT  A, B, C, D
```

时，计算机会依次发出多个问号，有几个变量就有几个问号。不过这几个问号不是同时出现的。它是先发出一个问号，你输

入一个数据并按了回车键之后再出现下一个问号“？”。这样你就可以依次将全部数据提供给相应变量。当所有的变量都被提供了数据之后，再按回转键，程序就会往下执行。这可以看一个例子：

```
10  INPUT  A, B, C
20  LET    D=B*B-4*A*C
30  PRINT  "D=" ; D
40  END
    RUN
    ? 4
    ? 9
    ? 2
    D=49
```

你还可将上例中输入数据的方法简化一下，即在输入第一个数之后不按回车键而是输入逗号“，”，每按下一个逗号之后，屏上就会出现一个新的问号，然后输入下一个数据，这样直到三个数据全输入后再按回车键，请看：

```
10  INPUT  A, B, C
20  LET    D=B*B-4*A*C
30  PRINT  "D=" ; D
40  END
    RUN
    ? 4 , ? 9 , ? 2
    D=49
```

在前一章我们学到了许多编程序的方法。现在，在掌握了一些语句的使用方法后，你已经可以写一些简单程序了。下面是第四章例2那个问题所需要的程序：

```
10  INPUT  A
20  INPUT  B
30  LET    W = A
40  LET    A = B
50  LET    B = W
60  PRINT  A, B
70  END
```

READ——DATA

这是读取数据的一对语句，READ语句和DATA 语 句必须连在一起使用。在一段程序中有一个READ 语句，就 必 须有一个或一个以上DATA语句与它对应。在很多 情 况 下，需要向计算机输入大量资料和数据，也就是说在某个程序中要对许多变量赋值，而且这些数据也都是已知的。在这种情况下使用INPUT语句就不太方便了。这时你可以用 一 组 更 有 效 的READ-DATA语句来提供数据，例如：

```
10  READ  A, B, C, D
20  DATA  11, 18, 14, 2
30  LET   E = (5 * A + 4 * B + 3 * C + 2 * D)
              /(A + B + C + D)
40  PRINT E
50  END
RUN
3.8445
```

READ是读数据语句，写在该语句中的是一个或 几 个 变量，上例中是A、B、C、D四个变量。这些变量之间必须用逗号“，”分隔开。READ语句中的变量可以是前面讲过的A～Z和A0～Z9之间的286个变量中的任一个。

DATA语句是与READ语句对应的。READ语句中有几个变量，DATA语句就必须有几个数据与变量对应。在DATA语句中的几个数据之间，必须用逗号隔开。DATA是非执行语句，它只是为READ语句提供数据。

当计算机在程序中执行到READ语句后，它就会自动地从DATA语句中去读取数据。当遇到第一个数据后，就将这个数据赋给READ语句中的第一个变量，然后依次继续将第二个、第三个……数据，分别赋给READ语句中相应的变量。从这里你会看出，DATA语句中数据的个数应该和READ语句中变量的个数一样多，否则就可能出现READ语句中的变量找不到数据的错误。另外你会看到，DATA语句中的每一个数据，都是要赋给READ语句中一个确定的变量的，相互不能混淆。这一点你在编写程序时应予以注意。因为当变量和数据很多时，容易出现差错。

READ-DATA语句的应用具有一定的灵活性，由于数据的使用和排列的关系，BASIC语言允许DATA语句写在READ之前，例如：

```
10  DATA  20, 30, 40, 50
20  READ   A, B, C, D
30  PRINT  A, B, C, D
40  END
RUN
 20  30  40  50
```

也可以将DATA语句写在程序最后，而不一定紧接READ语句，例如：

```
10  READ  A, B, C, D
20  LET   E = A + B + C + D
```

```
30 PRINT A, B, C, D, E
40 DATA 10, 20, 30, 40
50 END
RUN
10 20 30 40 100
```

有时READ语句中的变量很多，DATA语句中的数据也很多，这样就可以把一个READ语句分成几句READ语句来写。DATA语句也可分成几句来写，而且可以长短不一样，只需注意总的对应顺序就可以。在你编写程序时这一点可以灵活使用。请看下例：

```
10 DATA 10, 20
20 READ A
30 DATA 30, 40, 50
40 READ B, C, D, E
50 PRINT A, B, C, D, E
60 END
RUN
10 20 30 40 50
```

RESTORE ——恢复数据区语句

在一个程序中，有时一组数据需要反复使用，或者将一个数据提供给几个变量。为了解决这个问题，BASIC语言中不是将这些数据多次地用DATA语句写出，而是采取将一个DATA语句的数据恢复使用的方法。执行这个方法的就是RESTORE语句。我们知道，计算机在遇到READ语句之后，会从DATA语句中去找数据，读完一个再读下一个，好像有一个“指针”依次向后移动，指向READ语句要读的数据。RESTORE语句就如同把这个指针拨回到起始点，指向DATA

语句的第一个数据。无论READ语句读到DATA语句的哪一个数据，RESTORE语句都会命令后面的语句，从DATA语句的第一个数据开始再次提供数据。下面看一简单实例：

```
10  DATA  1, 3, 5, 7, 9
20  READ  A, B, C
30  RESTORE
40  READ  X, Y
50  PRINT  A, B, C, X, Y
60  END
RUN
 1   3   5   1   3
```

如果去掉第30句RESTORE，这段程序打印的结果就会变为：1、3、5、7、9。

以上介绍了提供数据的三组语句：LET、INPUT、READ-DATA和RESTORE。为了说明使用方法，所举的例子都是很简单的，你可以尽快地熟悉和掌握这些语句的用法。在以后你编写程序时可以灵活使用，编出更好的程序。

5-5 数学函数

前几节介绍的例题和程序都是很简单的，所处理的数据也很简单。但是计算机是具有很高的数学运算能力的。它的功能要比电子计算器还要强很多。

计算机的BASIC语言中规定了许多数学函数，这些函数可以写在程序语句之中，也可以直接运行，例如开平方函数SQR，从下面实例中你能看到这样两种用法：

```
PRINT    SQR(9)
```

```
3
10    LET A = 9
15    LET B = 36
20    LET A = SQR(A)
30    LET B = SQR(B)
40    PRINT A, B
50    END
RUN
 3          6
```

BASIC语言中的函数一般用大写英文字母表示，我们将BASIC语言中常用的数学函数列表如下：

语言	名称、功能	说明
SQR(X)	求X的平方根即$\sqrt{X}$。	X应大于等于零，(X≥0)
ABS(X)	求X的绝对值|X|。	
SIN(X)	正弦函数sinx	自变量X以弧度为单位。
COS(X)	余弦函数cosx	
TAN(X)	正切函数tgx	
ATN(X)	反正切函数$tg^{-1}x$	函数值单位为弧度$\left(-\frac{\pi}{2}<\text{函数值}<\frac{\pi}{2}\right)$
LOG(X)	以e为底的自然对数lnx	自变量X>0
SGN(X)	取符号 当X>0 SGN(X)=1 当X=0 SGN(X)=0 当X<0 SGN(X)=-1， 即取某数值的符号（正、负或0）	自变量为任意数
EXP(X)	求e^x (e=2.71828)	e为自然对数的底
INT(X)	取整函数，求不大于X的最大整数	例如：INT(9.8)=9 INT(-9.8)=-10
RND(X)	随机函数。产生0～X之间的随意整数	X值不可是负数，当X=0，RND=0～1之间的浮点数

几点说明：

（1）在上述函数中，自变量X外面的括号是必不可少的。

（2）自变量X可以是数值，也可以是已赋值的变量或算术表达式。

（3）RND(X)函数功能为：计算机在每执行一次RND(X)之后产生一个随机整数，这个数是任意的，不可预测的，它的范围在0～X之间。RND(X)在不同计算机中，规定有所不同，使用中请予以注意。

有了这些数学函数，就可深入一步进行一些较高级的数学运算。下面举几个例子，通过实例对上面表中的一些函数用法作实际了解：

例1
```
PRINT ABS (5 - 3 * 4)
 7
```

这里的X为5 - 3 * 4 =- 7，而ABS(- 7)之后得出“7”的结果，完成了对括弧中表达式取绝对值的操作。

例2
```
10  A = 18: B = 0: C = - 25
20  PRINT SGN(A),SGN(B),SGN(C)
30  END
RUN
 1      0      - 1
```

取符号函数SGN(X)的含义是：若X是正数则只取其正号以+1表示，一般省去+写作“1”；若X是负数则取负号，用“- 1”；若X = 0则无正、负可分，故标“0”。

例3
```
A = - 18.7: B = INT(A):PRINTA、B
 - 18.7          - 19
```

在对一些数据的分类中，有时需要将带有小数部分的数化为纯整数，应用INT(X)语言就可方便地实现。上例中对一数

“-18.7”取整后结果是“-19”而不是“-18”，你按INT(X)函数定义想一想对不对。

在输入BASIC语言的数学函数时要一个字母一个字母依次去按键盘上相应的字母键。例如SQR(A)就应顺序按[S] [Q] [R] [(] [A] [)]各键。但是有些简单的普及型计算机中，往往将数学函数功能定在某一个键上，作为该键的第二功能。这时，输入这些函数时，也可以不用依次按字母键，而是找到相应的函数键，照第二功能键按法按一个键就可以得到三个字母的数学函数，这就象计算器的函数键一样了。初学时可以采用这种方法，可以加快输入速度并避免按错字母。正规使用时应采用逐个字母输入方式。

5-6 程序流向控制语句

前面学过的程序，大都属于“直线”程序。这种程序是顺序地按语句标号，由小到大执行的。但是对计算机来讲，它的一些高超的运算功能，还主要表现在它能反复执行某一段程序，并且在每执行新的一次时，程序中变量的数值也随之变化，从而完成一些复杂的运算，代替人们不容易做的或计算量很大的运算。这一节中将介绍四个功能更强的语句，使用这些语句，你可编出更好的程序。

GOTO——无条件转移语句，即有“转向”之意

FOR—NEXT——循环语句

IF…THEN——条件转移语句

GOSUB-RETURN——转子程序语句

一、GOTO语句

BASIC语言的程序，一般情况下都是按照标号由小到大顺序执行的。但是当遇到GOTO语句时，程序就要转移到GOTO语句中所指定的那一条语句去执行。例如：

```
10  A = 5
20  PRINT  A,A*A
30  GOTO  10
40  END
RUN
5            25
5            25
5            25
⋮            ⋮
⋮            ⋮
```

你将会看到这段程序运行后，显示器上不停地连续显示 5 和25这两个数。因为程序执行完第20句之后，到了第30句遇上无条件转向第10句的语句，所以程序又执行第10句、第20句，这样循环往复不停。这就是GOTO语句所造成的结果，它告诉计算机不要进入第40句结束，而要转向第10句，重新运算。为了结束这种不停的运算，在COMX-35机中可按下[ESC]键。该键的功能是中止程序的执行，并准备接受新的BASIC指令。其它型号计算机中，也有的用下述语句结束循环运算。

BREAK

当出现上述程序运行不止的情况时，你可按BREAK键(中断键)，程序就会停下来，并且屏幕上会告诉你停在了哪一句。

CONT

可是，当你还想再看一下上述程序运行情况和结果时，你可以键入CONT（继续），计算机就会接着从程序先前的停止

处继续运行。

二、FOR—NEXT循环语句

在BASIC语言中除了有GOTO定向转移的语句之外，还有一种可以自己确定在一定范围内循环的语句，这就是FOR-NEXT语句。它不仅可以完成循环，而且可以确定循环的次数和在循环中自动改变变量的数据，因而它对编制解决重复计算问题的程序是十分方便的。我们先看一个实例，即计算并打印出1、3、5、7、9五个奇数的平方值：

```
10  FOR  X=1  TO 9  STEP 2
20  Y=X*X
30  PRINT  X, Y
40  NEXT  X
50  END
RUN
 1            1
 3            9
 5           25
 7           49
 9           81
```

FOR-NEXT语句的标准形式是：

```
FOR  X=A  TO  B  STEP  C
[循环体语句]
NEXT  X
```

下面分别说明一下这种语句形式，循环语句的第一句要比前面见到的语句长一些，它由三个“单词”FOR、TO、STEP组成，中间还夹有变量和数值：

X——循环变量。一般它在循环过程中变化，它也可以使用其它英文字母，如I、J、K等表示。

A——循环变量X的初始值。

B——循环变量X的最终值。

C——步长，即循环变量每次增加的数量。

FOR X = 1 TO 9 STEP 2语句的意义是：循环变量X从X = 1开始变化，每循环一次变化一个步长（STEP后面的数值），一直变化到超过终值（TO后面的数值）为止。此例中步长为+ 2，终值为9。

NEXT X 语句是指“执行X的下一个值”，此语句对循环语句来说也是至关紧要的。没有NEXT语句，循环语句就不知道执行到何处返回。NEXT语句与FOR语句是一、一对应的，而且NEXT语句中的变量必须与FOR语句中的变量一致（即是同一变量如X），否则就是错误的。

循环语句的初值A、终值B和步长C可以是常数，也可以是变量或表达式。A、B的值可以是正数、负数或零。而C的值不能为零，否则循环就会以“步长= 0”无休止地不变化地进行下去，就达不到真正循环的意义了。当步长C的值为1时，FOR循环语句中步长这项即“STEP C”可以省略不写。下面一个例子是打印从1—10平方表的程序，此例中步长为1，故省略不写：

```
10  FOR  X = 1  TO  10
15  Y = X * X
20  PRINT  X, Y
30  NEXT  X
40  END
RUN
```

1	1
2	4
3	9
4	16
⋮	⋮
9	81
10	100

下面再看一个实例：求10!，即1×2×3…×10

```
10  LET  T = 1
20  FOR  I = 2  TO  10
30  LET  T = T*I
40  NEXT  I
50  PRINT  T
60  END
RUN
```

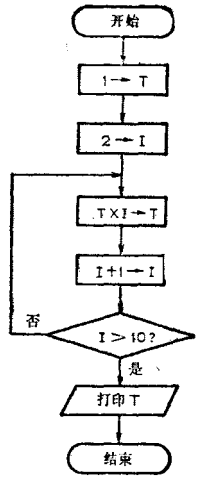


图 5·6·1 流程图

在BASIC程序的一个循环体中是允许再有小的循环语句的，这称为多层循环。你如果编写这样的程序时，一定要正确掌握它的使用方法。请看图5·6·2的程序。如果将40句写成“40 NEXT A”，将50句写成“50NEXT B”，这时两个循环发生了交叉（如虚线所示），循环就无法进行。就是说，内外循环线不能交叉。这是编程中的一条原则。

三、IF…THEN条件语句

这条语句在BASIC语言中也是很重要的，它赋予了计算机进行判断功能。它的含义是：“如果……那么……”。现在先看一个简单的例子：统计一个班学生考试成绩，找出90分以

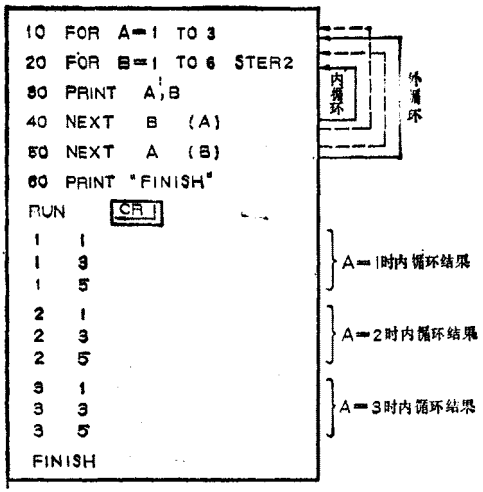


图 5·6·2 循环语句的内循环和外循环

上的学生，并打印出它的学号及成绩。设N为学 号， D 为分数，程序如下：

```
100  INPUT  N, D
110  IF  D<90  THEN  GOTO  100
120  PRINT  N, D
130  GOTO  100
RUN
?   7
?   90
 7        90
?   10
?   80              (由于D<90，故不打印)，直接转
?   15              第100句，问下个学号)
?   95
```

```
 15        95
 :         :
 :         :
```

这段程序运行情况是这样的：计算机在执行了INPUT语句后，就会打出问号，这时你可用键盘输入学生的学号及成绩，然后IF…THEN语句判断成绩D的值是否小于90，若小于90，则转到100句，重新输入下一个学号及成绩，例子中第二个学生成绩≥80分，就没有打印出来。如果成绩不低于90分，就打印出学号和成绩分数。

条件语句的形式为：

IF 条件比较式 THEN 语句

它的定义是：如果条件比较式表示的条件成立，就执行THEN后面的语句，否则就顺次执行下面一条语句。BASIC语言规定条件比较式中可以使用以下比较符号：

=	等于
< >	不等于
<	小于
>	大于
>=	大于或等于
<=	小于或等于

下面是条件比较式的几种形式：

（1）一个变量与一个数值的比较：

```
10 IF A>3 THEN GOTO 60
```

（2）两个变量进行比较：

```
40 IF X<Y THEN GOTO 100
```

（3）一个变量与一个表达式比较：

```
120 IF T<A*2+5 THEN GOTO 70
```

（4）两个表达式比较：

```
60 IF 3*A*B<>5*LOG(A) THEN GOTO 80
```

条件语句例子如下，理解它的含义也并不难：

```
10 IF A=B THEN PRINT A:GOTO 100
20 PRINT B
    ⋮
```

第10句条件语句内容为如果A = B，则打印 A 的值，再转向100语句去执行；如果A不等于B，那么往下执行第20句语句，打印出B的值来。

IF…THEN语句在不同的计算机中用法和规定在细节上都有所不同，使用时请参阅你所用计算机的使用手册中这方面内容。

你可能还记得第四章流程图实例中的第 3 例，那是一个计算铁路行李运费的问题，现在用程序语句来实现它：

```
10 INPUT G
20 IF G>50 THEN GOTO 50
30 LET Y=0.15*G
40 GOTO 60
50 LET Y=0.15*50+0.25*(G-50)
60 PRINT "YUN FEI=", Y
70 END
```

这里的“YUN FEI”是“运费”两字的汉语拼音。

四、GOSUB-RETURN 转向子程序语句

在一个实际应用的程序中，往往有一段程序要多次被用到，如果要用“直线”程序方法，这段程序要多次地写到整个

程序里。显然这样做比较费事。为此，程序设计中使用了子程序的设计方法，将整个程序作为主程序，而将其中经常要用的语句相同的几段小程序合为一段，把它作为子程序。BASIC语言规定GOSUB为“转向子程序”语句，例如“30 GOSUB 100”是转子程序语句。计算机遇到这条语句后，就转向GOSUB后的第100语句去执行。

GOSUB语句类似GOTO语句，但不同于GOTO语句，关键之处在于使用GOSUB语句转向子程序时，计算机会自动记住程序是从何处转向子程序的，当子程序执行完了之后，遇到RETURN语句（返回语句）时，再返回到主程序转出的地方，继续执行主程序。若用GOTO句就无此功能。

RETURN语句是子程序返回语句，它写在每段子程序的最后，是必不可少的。下面先看一个例子。此例在实际应用中并不一定完善，但它对于你理解为什么要使用子程序和怎样编写子程序是有帮助的。请注意GOSUB和RETURN的用法。

例 设一个圆的半径为R，求R = 1、R = 2、R = 5、R = 9时圆的面积。

```
10  R = 1
20  GOSUB 100
30  R = 2
40  GOSUB 100
50  R = 5
60  GOSUB 100
70  R = 9
80  GOSUB 100
90  END
```

```
100  S = 3.14159 * R * R
110  PRINT "R = "; R
120  PRINT "S = "; S
130  RETURN
```

子程序

```
RUN    CR
R = 1
S = 3.14159
R = 2
S = 12.5664
R = 5
S = 78.5398
R = 9
S = 254.469
```

这段程序是这样执行的：当执行完第10句之后，在第20句遇到GOSUB100，程序立即转向第100语句去执行。从100句到130句是一小段完整的子程序，它完成求圆面积$S=\pi R^2$的运算。求完圆面积之后，关键的是第130句RETURN，它将程序返回到刚才转来语句的下一条语句，即第30句，这样程序又从30句开始继续往下执行……。

在这个初步的实例中你可以看到，转子程序语句应该说是由两条相互呼应的语句组成。一条是转向某某子程序语句的GOSUB语句，另一句是返回主程序语句RETURN。

如果将主程序写在一起，另将子程序写在一起，主程序和子程序的关系可用图5·6·3表示：

学到这里，你一定会想到在子程序中可不可以再包含另一个子程序呢？回答是肯定的，它们的关系如图5·6·4所示。

在这种子程序中套子程序的使用中，不要忘记在每一个子

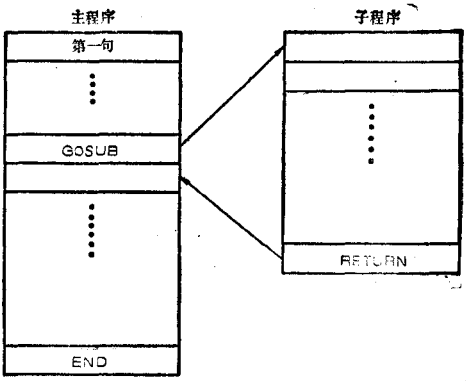


图 5·6·3 主程序和子程序关系图示

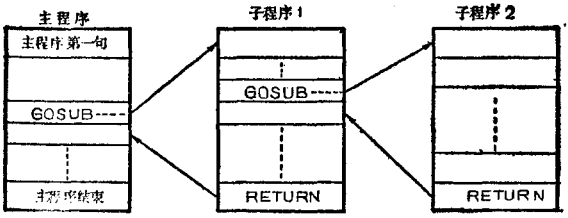


图 5·6·4 子程序嵌套图示

程序的末尾写上RETURN返回语句。子程序中再套子程序在BASIC术语中称为子程序的嵌套。使用子程序的嵌套，也是编程序的一种技巧。但是这种嵌套不是无限制的。BASIC语言规定子程序嵌套的层数（即可嵌套几层子程序）是有限制的，具体规定几层可参阅你的操作使用手册。

5-7 字符串变量

现在你已经对A、B、C8、D9这种形式的变量比较熟悉

了，它们都是代表一定的数值的，如A = 35、 B = 72 + 3 * 4 、C8 = 95、D9 = A + B + C8。但是在BASIC语言中还有表示其他内容的变量。这一节介绍的变量就是代表几个字母或字符的，所以它被称为字符串变量。BASIC语言规定，字符串变量用一个大写英文字母（A～Z中任一个）加上一个专用符号“$”表示，例如A$、B$、C$…Z$等。

字符串变量在程序中一般是这样出现的：

```
10  LET  A$ = "ABC"
20  LET  B$ = "DEF"
30  LET  C$ = "GHI"
40  LET  D$ = A$ + B$ + C$
50  PRINT  D$
RUN
ABCDEFGHI
```

在这个短小的程序中你可以看到字符串变量的一般用法，例如第10句中的A$就是字符串变量、引号中的ABC就是字符串，ABC两边的引号是必不可少的。BASIC语言规定，字符串必须用引号引起来。因为引号是作为字符串的分界号，即用来表示字符串的开始和结尾。再看程序第40句中的A$ + B$ + C$，可见字符串变量也是可以进行运算的，运算的结果是将ABC与DEF和GHI加在一起成为一个大一些的字符串 ABCDEFGHI，并且将它赋给D$。可以看出字符串是可以相加的。

BASIC语言规定：（一）一个字符串由若干字母、数字、特殊字符和空格组成（注意空格也算一种字符）。字符串在语句中出现时，其两边必须加引号，但引号不算字符串内容。一般字符串的长度有一定的限制，如不允许超过127 或 256 个字符，使用中一般不会超出这个范围。

（二）字符串变量由A～Z中任一个字母和“$”符号组成，即A$、B$、……X$、Y$、Z$共26个。如：D$ = “BEIJING”

（三）输入字符串变量和输入简单变量一样，可以使用LET、READ-DATA、INPUT三种语句，而且在一个提供数据的语句中，允许将A、B、C6等变量与A$、B$等字符串变量写在一个语句中。例如：10 INPUT A，B$，C$，C6，D当然，计算机执行到这条语句时，就要依次发出五个问号。你在回答时就应该将字符串与字符串变量对应，而将数值与简单变量对应。同样，在READ-DATA语句中也可有这两种变量，只要将每一个变量与它的内容对应好就可以了。

（四）字符串之间可以进行比较和进行相加运算。字符串和字符串变量的引入，一个重要的目的，就是要对一些字母名称进行检索和比较，以达到查找和分类的目的。例如假设一个班上有张、王、李、赵四个同学，现在要查找王同学的成绩并打印出来，用一个简单程序可以完成：

```
10 FOR I = 1 TO 4
20 READ A$,A
30 IF A$ = "WANG" THEN GOTO 60
40 NEXT I
45 PRINT "NOT"
48 GOTO 70
50 DATA "ZANG", 80, "LI", 96, "WA
   NG", 93, "ZHAO", 92
60 PRINT A$, A
70 END
RUN
```

WANG 93

字符串之间不仅可以比较是否相等，而且可以进行不相等比较，用于字符串的比较符号同样有：

=	等于
< >	不等于
>	大于
<	小于
>=	大于或等于
<=	小于或等于

参与比较的两个字符串如何确定大小呢？原来在BASIC语言中已经将所有符号、数字、字母以及空格，都根据ASCII标准进行了编码，也就是将每一个字符编成了相应的一个7位二进制数，或者0～127的十进制数（请参阅本书后面附录中ASCII编码表）。你可以找到每一个字符的代码数值。但在这一节，只需要比较字符大小，你只要记住字符大小的次序就可以了。下表是按由小到大顺序排列的，在每一行中表示左边的比右边的小；上一行最右侧一个比下一行最左侧的小。

空格	!	"	#	$	%	&	,	(
)	*	+	,	-	.	0	1	2
3	4	5	6	7	8	9	:	;
<	=	>	?	@	A	B	C	D
E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V
W	X	Y	Z	↑				

从表中看出“空格”最小，“↑”最大。两个字符串相比，是将两个字符串的最左侧字符相比，若字符相同，则从左到右逐个相比，直到两个字符串中相对应的字符不相同为止，那么就

以这个不同的字符比较结果为准。例如：

```
10  A$ = "ABC"
20  B$ = "ABD"
30  IF B$>A$THEN PRINT "B$"
40  END
RUN
B$
```

这里A$与B$字符串的第一、二字符都是A、B，故只能以第三个字符为准了。A$的第三个字符是C，而B$的是D，由于D>C，故B$>A$。

字符串的运算只限于将几个字符串相加，字符串之间不能相减或乘、除，且相加是指将字符串前后接起来之意，并不是其大小相加，例如：

```
10  A$ = "I AM"
20  B$ = " A "
30  C$ = "STUDENT"
40  D$ = A$ + B$ + C$
50  PRINT  D$
RUN
 I  AM  A  STUDENT
```

字符串函数

由于字符串变量的引入，给计算机程序设计带来了许多方便。字符串变量的使用有它自己的特点，因此BASIC语言又规定了一些字符串函数，用于进行字符串变量的运算和处理。字符串函数在不同型号计算机中规定也有所不同，下面介绍常用的几个：

LEN（字符串变量）

这个函数的含义是令给出所指定的字符串中字符的个数，所以LEN叫字长函数，它的后面一定跟一套括弧，括弧中有一个字符串或字符串变量（必须预先有定义）。例如：

```
10  A$ = "WANG"
20  B$ = "WA  NG"
30  PRINT  LEN  (A$)
40  PRINT  LEN  (B$)
50  END
RUN
 4
 5
```

从程序中看到A$中有4个字符，而B$中有5个字符，注意，其中包括一个空格。

MID$ ( )

此语叫子字符串，在它后面必须跟一套括号，中有一个字符串变量，然后是两个数字。这个函数的功能是这样的，它从括号中，已经定义过的字符串（如A$）中取出某一段相邻的字符，组成一个新的字符串。由于这个新的字符串，是从一个原有的字符串中取出一部分组成的，所以它也被称为原字符串的子字符串。

执行MID$函数时，取出所指定字符串的一部分，这一部分是由括号中前后两个数规定的，其中第一个数规定了从左算起第几个字符作为子字符串的开头；而第二个数则规定了子字符串中字符的个数。如果不写第一个数，则表示从头取起；如果不写第二个数，则所有剩下的字符全取。例如：

```
10  A$ = "EXAMPLE"
20  B$ = MID$(A$, 3, 2)
```

```
30  PRINT  B$
40  END
RUN
AM
```

上例中B$=MID$(A$,3,2)表示：从A$中第3个字符开始取2个字符赋给B$，结果使B$=AM，AM就是字符串EXAMPLE的子字符串。也有的机器也有用“PRINT字符串变量（数字TO数字）”的形式。

ASC（ ）

这个函数求出其后括弧中字符串的第一个字符的ASCII编码数值，并以十进制数形式给出。前面讲过，BASIC语言中每个字符都用一个数值编码（一般都采用美国ASCII标准）。例如字母A的编码是65、B是66、C是67……。数字0的编码是“48”，1是“49”，2是“50”……，详细的可查附录ASCII编码表。例如：

```
A$ = “CHINA”
B=ASC(A$)
```

那么变量B的值就为67，67就是字母C的十进制ASCII码数值。下面在举一个程序例子：

```
10  A$ = “TEST”
20  FOR  A=1  TO LEN(A$)
30  PRINT  ASC  (MID$(A$,A))
40  NEXT  A
50  END
RUN
84
69
```

```
83
84
```

运行后打出的四个数字，分别是TEST四个字母的编码数值。

CHR $ ( )

这个函数与上面ASC函数是相对应的，其功能正好相反，CHR $是将其后面的括号中的各数值（ASCII码）译为相应的字符。

例如

```
PRINT  CHR $(66)
$
```

就是将编码数值66译成了字母B。

关于字符串函数，在不同的计算机中规定略有不同，上面介绍的是常用且基本相同的几种。当你拿到你所用的计算机时，查阅它的使用手册，你会找到它规定的全部字符串函数。

5-8 数组和DIM语句

数组和数组元素

数组也称为带有下标的变量组，简称下标变量组，也有叫阵列或矩阵的。

前面学过了BASIC语言中的变量A、B、D7、E6等等，它们都是简单的变量。这一节介绍下标变量。即形式如A(3)，B（4），D(3,4)，这样形式的变量。它们与简单变量A 3 、B 4 等很相象，但它在每个字母之后都加了一个括号，括号中的阿拉伯数字称为下标。因为计算机屏幕显示的每一行中不能分出上、下，所以下标是与字母并列写的，加上一个括号就表示它是下标了。例如A(3)就是表示我们日常书写时的A_3。

数组和数组元素也是根据数学计算和编写程序的需要而提

出来的。例如学校中一个班有50个学生，现在要检查他们的身高和体重并算出其平均值，这样就会出现150个数据或变量(学号50个，体重50个，身高50个)。若只用以前介绍过的简单变量来表示，就会出现不够用的情况。而且变量和每个人又很难一一对应，你分不清哪个变量是代表哪个同学的身高和体重，显然这是很不方便的。因此你会想到，若用H来代表身高，而用每个学生的学号1～50代表每个同学，将变量写成H(1)、H(2)、H(3)…H(49)、H(50)的形式，这就解决了用简单变量表示的不足。同样，可以用W代表体重，那么50个同学的体重可以用变量W(1)、W(2)…W(50)来表示。

很显然，数组就是一组数，而且这组数据有相同的特点和属性，此特点可以用一个字母来表示。上例中的H(1)、H(2)…H(50)就是一个数组。H叫做数组名，数组中的每一个变量都称作数组元素，如H(5)、H(12)都是数组元素。一般数组元素都是有顺序地排列着的，这一点和简单变量不同。

数组元素一般有A(3)、B(1)、C(K)、D(I+5)等形式，它由一个大写英文字母、括号、下标三部分组成。BASIC语言规定，括号中的下标可以是常数、变量或表达式，例如A(3)、C(K)、D(I+5)都是正确的。但应该注意，下标中如果有变量如K、I等等，它们应事先被赋值。下标的值还必须大于0或者等于0，不能是负数，所以若写出A(-3)、B(-5)就是错误的了。

一维数组

上面提到的A(3)、B(1)等数组元素，它们括号中的下标只有一个数(包括一个表达式如I+5的值也是一个数)，这样的数组称为一维数组。而如D(3,4)、C(4,3)的则为二维数组。它的括号中有两个下标变量，而且二者之间加

了逗号。二维数组将在后面介绍，这里先讲一维数组。

在讲数组的应用之前，必须先讲一讲另一个语句DIM。这是一个数组定义语句，即用以定义某一个数组中的元素有多少个。它的形式是：　　DIM　字母字符（数字）

例如：DIM　A（50）

就是表示以A为名的数组中有51个元素，依次为A（0）、A(1)、A（2）……A（50）。在使用中，数组A的元素只能在上述51个中选取，不能超出此范围。

在同一个DIM语句中，可以定义一个或数个数组，例如：

DIM　A（50），B（40），C（10）

下面举一个应用数组来编程的例子。

统计计算某一班50名学生的身高的平均值，并打印出来。我们可用下述程序：

```
5  DIM  H (50)
10  T = 0
20  FOR  I= 1  TO 50
30  READ  H (I)
40  T = T + H (I)
50  NEXT  I
60  T = T/50
70  PRINT  "MEAN =" ; T
80  DATA  1.41, 1.43, 1.38……1.45
                          (50个数值)
90  END
RUN
MEAN = 1.40
```

二维数组

顾名思义，二维数组与一维数组是类似的，它的形式为A（3、4）、B（2、2）……。一维数组用来表示一列性质相同的变量是方便的，但是在实际中，还有另一种有规律的数字或变量，它们的排列好象一个方阵：

△△△△
△△△△
△△△△

比如仓库中存放的货物就是横成行，竖成列的，为了表示这“方阵”货物的种类和重量，仅仅用一维变量就显得太不够用了。仿照数学中“矩阵”的方法，BASIC语言规定了二维数组变量，上例中的三行四列的变量矩阵可以用一个二维数组A来表示：

A（1，1）A（1，2）A（1，3）A（1，4）
A（2，1）A（2，2）A（2，3）A（2，4）
A（3，1）A（3，2）A（3，3）A（3，4）

这样一个二维数组A就很简便地表示了十二个变量。当二维数组元素的数量大到成百上千的时候，这种表示方法就更显示出优点。二维数组变量的一般形式为A（m，n），第一个下标m，称为行下标，第二个下标n，称为列下标。

二维数组在使用中，也存在着必须先定义的问题，例如：

DIM A（3，4）

当然，在很多BASIC语言版本中，一维数组中若数组元素少于11个，则定义语句DIMA（10）可以省略不写；二维数组中若数组元素少于121个，则数组定义语句DIM B（10，10）也可以省不写。但是，为了养成事先定义存贮空间的编程习惯，避免未事先定义而发生程序中的错误，希望你还是在每次使用数组变量之前，先用DIM语句定义一下。

在有的计算机BASIC语言中，还提供了处理字符串的能

力，规定了字符串数组，因此可以用A$（N）、B$（N）等来作为字符串数组变量，其中N是正整数或表达式，其最大值是255。有一点需要注意的是，字符串数组不需要预先规定存贮空间的大小，而是每产生一个字符串，存贮器就自动分配一次。因此在使用上就比较方便，A$（3），B$（7）等的使用就如象C5、D7等简单变量的使用一样，请看下例：

```
10  INPUT  B$
20  A$（1）= “CONTINUE”
30  A$（2）= “PHASE 1”
40  A$（3）= “PHASE 2”
50  IFB$=A$(1)+A$（2）GOTO
    100
60  IFB$=A$（1）+A$(3)GOTO
    200
70  GOTO  10
```

上面的程序将停在第10句等待输入一个字符串，尔后这个字符串就存放在B$中，将B$分别与A$（1）+A$（2）和A$（1）+A$（3）比较，按照相同与否，程序转到相应的地方去执行。

最后，以一个实例结束这一节内容。初中代数中有解二元一次联立方程组的问题，请看用BASIC语言程序如何解所有这种类型的题：

例 解二元一次联立方程组

$$\begin{cases} 4X_1 - 3X_2 = 6 \\ -2X_1 + 4X_2 = -5 \end{cases}$$

这种方程组的一般形式是

$$\begin{cases} a_{11}x_1 + a_{12}x_2 = C_1 \\ a_{21}x_1 + a_{22}x_2 = C_2 \end{cases}$$

$$x_1=\frac{c_1a_{22}-a_{12}c_2}{a_{11}a_{22}-a_{12}a_{21}} \qquad x_2=\frac{a_{11}c_2-a_{21}c_1}{a_{11}a_{22}-a_{12}a_{21}}$$

根据上述数学原理，可以编写出解二元一次方程组的 BASIC语言程序：

```
10  FOR I = 1 TO 2
20  FOR J = 1 TO 2
30  PRINT "A (" ; I ; ", " ; J ;
    ") =" ;
40  INPUT A (I, J)
50  NEXT J
60  NEXT I
70  PRINT "C (1) , C (2) " =;
80  INPUT C (1) , C (2)
90  D = A (1, 1) * A (2, 2) - A (1,
    2) * A (2, 1)
100 X1 = C (1) * A (2, 2) - C (2) *
    A (1, 2)
110 X1 = X1/D
120 X2 = C (2) * A (1, 1) - C(1) *
    A (2, 1)
130 X2 = X2/D
140 PRINT "X1 =" ; X1, "X2 =" ; X2
150 END
```

程序运行情况如下：

```
RUN
A (1, 1) =?  4
```

```
A（1，2）=？－3
A（2，1）=？－2
A（2，2）=？ 4
C（1）、C（2）=？6？－5
X1=0.9
X2=－0.8
```

方程的系数和常数由键盘输入

你可能觉得这个程序长一些，认为解一道二元一次方程组的题，不用费这么多事就可做出。但是你仔细思考一下会发现，这个程序提供了解二元一次方程组的一个通用的解法。无论你遇到多少个这类应用题，尽管方程式的各系数和常数千变万化，但计算机解题程序不变。你只要输入不同数值，几秒钟就会算出答案。

5-9 BASIC语言的另一些命令和语句

以上几节讲述了BASIC语言的基本语句和命令，这些内容对于学习使用计算机的人来说，是应该必须掌握的。本节和下一节的内容，相对地讲，在你编写程序中要用得少一些。初学者可以先不看这两节，待你学完本书的其他章节之后，再来读这两节，会使你对BASIC语言有更进一步的认识。

这一节介绍的语句和命令分为三种情况：其一是在程序中使用不太多的命令和语句。普及型计算机种类很多，功能也不完全相同，所使用的BASIC语言版本也不一样。因此这些种计算机除大部分主要语句和命令相同之外，其它个别语句在功能上和格式上都不一样。对这些语句和命令，这里只做简单的解释。开始学习时也不要将其作为重点，待你将前几章节内容掌握好之后，你可再来仔细思考它们。

其次是前面几节没有讲到的命令或语句，虽然这些不象前几节介绍的内容那么重要，但在程序中也有时会遇到的。

第三种情况是前面讲过的语句或命令。为了突出重点，前面只讲了它们的基本形式和用法，在这一节再补充全面些，以符合有些型号计算机的规定。

EXIT（表达式）

这个语句是无条件地转移语句。程序中可用它来提早跳出FOR-NEXT循环或子程序。有一点应注意的是，如果子程序或FOR-NEXT循环是多层嵌套的，则EXIT语句所规定 转移控制的行号（即EXIT语句中表达式所指行号），应位 于下一层嵌套之内，也就是说EXIT语句只能跳出一层，而不能 一下子从多层嵌套中跳出来。例如：

```
 5  INPUT  B, C
10  FOR  I = 1  TO  10
20  FOR  J = 1  TO  5
30  A = B + C
40  IF  A = 15  THEN  EXIT  70
50  PRINT  A
60  NEXT  J
70  PRINT  A
80  NEXT  I
90  END
```

A≒15时EXIT语句不起作用，那么就执行第50行，就要打印10×6=60个A，若A=15，则EXIT语句起作用，跳出小循环（J=1TO5），直接执行第70行指令，就只打印10次A了。可以看到EXIT语句只跳出第一层循环，而没有跳出第二层循环。

EXIT语句可用来代替GOTO语句，但它与GOTO语句不完全一样，EXIT语句执行时可清除FOR-NEXT或子 程序调用所产生的所有变更了的堆栈指示器。

WAIT（表达式）

暂停语句，它提供一种在程序执行中暂停的方法。暂停的时间长短与括号中的数值成正比。当括号中的数值等于1时，暂停的时间约为6.4毫秒。下面是WAIT语句在程序中 的用法：

```
10  INPUT  A$ (1)
20  PRINT  "MESSAGE  IS" ; A$ (1)
30  WAIT (500) :CLS:GOTO  10
```

这个程序能在执行CLS语句（清除显示屏）之前有一 段 停 留时间（500×6.4＝3200毫秒＝3.2秒）可看到信息，然后GO-TO 10，再继续执行。

CLS

部分清除显示指令语句，其意思是，清除现有 光标 位置以下的显示屏。注意仅仅是将显示的字符清除掉，并 不 是在存贮器中清除掉。

CPOS（表达式，表达式）

光标位置语句。它将光标移动到括号内两数 所 指定的屏幕上的位置，第一个数值规定其行位置（从0到23），第二个数值确定其列的位置（从0到39）。CPOS语句 与 CLS语句一起使用，可清除显示屏的一部分；与PRINT一起使用可将信息或标题从屏上指定的位置开始显示。例如：

```
10  CPOS (11,  0)
20  CLS
RUN
```

这里是将第11行，0列开始的屏下半部分清除掉。要 清 除整

个显示屏，可以打入CPOS（0，0）:CLS命令。

与PRINT配用时的例子是：

```
10  CPOS (11,12)
20  PRINT "THIS IS A TEST"
RUN
```

这时将在显示屏的中间显示出THIS IS A TEST字样。

CLD

清除数据语句。它执行时清除所有的字符串和数组，它是通过重新设置字符串和数组指示器到原始状态来实现的。

LIST

LIST语句之后若不加任何数值或式子，将列出用户编入的全部程序。如果跟着给出一个数据，则仅列一行，其行号就是该数据值。如果给出两个数（其中必需用“，”号隔开），则列出从第一个数所指定的行到第二个数所指定的行之间的所有程序。

NEW

该命令将计算机内以前存的程序和变量全部清除，并预置所有的字符串和数组指示器。

RUN

这个命令告诉计算机从最小行号开始并按行号顺序逐行执行。

如果在RUN后面紧跟一个数，那么这个语句从该数所指定的行号开始执行。这种RUN语句不清除数据，如果你的程序需要带有以前产生的数据，则可使用这种语句运行这段程序。

REM

注释语句。它一般放在一段程序的开头。整个这一行在程

序执行过程中不予理睬，仅在执行 LIST 命令时才列在显示屏上作为一段程序的说明性注释。

DEG

这个语句规定sin、cos等三角函数的计量单位为角度。一般计算机BASIC语言预置状态是弧度制，所以要使用角度制时，应先使用DEG语句。

RAD

这个语句将三角函数的计量单位改为弧度。它的功能正好和DEG的功能对应。

DEFINT

定义整型变量语句。DEFINT之后如果不带变量名，执行时所有变量（A—Z）和所有数组都选做浮点型。如果在DEFINT语句中包含有变量名，则从英文第一个字母 A 起到此变量名（包括此变量名）的所有变量和数组都取为整形。例如。

```
DEFINT D
```

就定义了以A、B、C、D为名称的变量和数组为整形。由于NEW语句总是将所有变量恢复为浮点型，因此如果在程序中要使用整形变量，就应提早在程序中出现DEFINT语句。

TAB（表达式）

打印格式语句。它只能在PRINT语句中才可使用，它将光标移到由括号内的值确定的水平位置上，光标的新位置是从本行最左边第 0 列开始数起，定在由括号内的值确定的那一列位置上。请看几例：

```
PRINT TAB (10) ; 1
```

将在第10列打印出 1 这个数。

```
PRINT TAB (10) ; "A" ; TAB (20) ; "B"
```

将在第10列打印出A，在第20列打印出B。

```
PRINT  TAB (A) ; "* * *"
```

将从第A列开始打印出三个星号“* * *”。

TAB语句对打印图形是很有用的，很多人喜欢用它来打印动物图案和欢迎标语，也可用它打印函数的曲线图形。你可使用TAB编一段程序，这段程序运行后，计算机就会打印出你所设计的图形。下面有一段实际程序，请你用计算机运行一遍，看看会出现什么图形。

```
10  FOR X = -3.14159 TO 3.14159 STEP
    3.14159/18
20  PRINT  INT (X*180/3.14159) ;
30  PRINT  TAB (22+10*SIN (X) ) ; "*"
40  NEXT X
50  END
```

5-10 用BASIC语言编写和调用机器语言程序

BASIC语言属于高级语言，而计算机本身只能识别二进制编码即机器语言。那么，你一定会想到，前面许多用BASIC语言写的程序，计算机是怎么识别和执行的呢？这是计算机的设计者事先做了工作的。他们将一个用机器语言写成的称作“BASIC语言解释程序”的程序，预先放入计算机的只读存贮器（ROM）之中。设计者还安排好让计算机一开机（或者一复位）就先去执行这段程序。那么这段解释程序就会帮助计算机正确地识别尔后到来的所有BASIC语言命令、语句和程序，并能正确地执行它们。这就使计算机具有了用BASIC语言工作的

能力。

明白了这一点，你可能会想到，BASIC语言中有没有直接编写和运行机器语言程序的命令和语句呢；回答是有的，在大多数计算机BASIC语言中都具有这种功能。下面作一简单介绍：

PEEK（某存贮单元地址码）

这是个取数命令。括号内的数码，是指所要调取的数据的存贮单元的地址码。地址码取十进制数0—32676。一般普及型计算机的RAM并不很大，大约不超过32K字节。

计算机存贮单元存放八位二进制数，使用了PEEK句后你就可以读出存贮单元的内容。八位二进制数的范围是从00000000到11111111，对应的十进制数是0—255。BASIC语言中用PEEK读出的数是用十进制数表示的。PEEK可以这样使用：

```
30 A = PEEK (28672)
```

这条语句会把地址码为28672的存贮单元中内容取出并赋给变量A。

POKE（存贮单元地址码，存入内容）

这个叫存数命令，是将括号内逗号后的数码写入逗号前的地址码所指定的存贮单元中去。它与PEEK正好是反义。写入存贮单元的一般只能在0—255范围内的二进制数字。下面是一个使用PEEK及POKE语句的例子：

```
10  A = 15
20  POKE (29000, A)
30  B = PEEK (29000)
40  PRINT B
RUN
```

15

有了这个语句，如果你又学会了所用计算机的机器语言，你就可以用机器语言编写程序，用POKE写入存贮单元。这段写好的机器语言程序如何去执行它呢？不同的计算机规定了不同的执行语句。为了简明地讲清楚用法，这里仅举LASER-310计算机中的USR（X）命令为例，看一看计算机怎样运行机器语言程序。

USR（X）

这个函数是使用自变量x调用机器语言（汇编语言）子程序。这个子程序可用POKE事先写入。

例如　　110 A = USR（B/2）

当这个函数执行时，计算机会从地址为十六进制数的788E和788F（十进制30862和30863）这两个存贮单元中，读出两个字节数据，并以这两个数作为机器语言子程序的开始地址。因此，在执行USR（x）命令之前，用户必须用POKE将机器语言子程序存入可以使用的存贮单元中。当然这段子程序存入后，它的首地址也就确定了。你可将这个首地址放入788E和788F（即30862和30863）两个单元之中。USR（x）中x的数值也可用于机器语言子程序，它的数值是存放在十六进制数7921和7922（即十进制31009和31010）两个内存贮单元之中。

*　　*　　*

本章的内容到此就介绍完了。在结束之前再来看看第四章的程序框图，那是一些很好的实例，请你按照本章介绍的BASIC语句去实现它。下面几个程序，供你编写程序时参考和对照。下面例子的编号，与第四章例题编号一致。

例4　求100个数字的总和

```
10  LET W = 0
20  LET I = 1
30  INPUT A
40  LET W = W + A
50  LET I = I + 1
60  IF I <= 100 THEN 30
70  PRINT W
80  END
```

例5 求数值在10～100之间的若干个数的连乘积。

```
10  LET W = 1
20  LET I = 0
30  INPUT A
40  IF A < 0 THEN 80
50  LET W = W * A
60  LET I = I + 1
70  GOTO 30
80  PRINT W, I
90  END
```

例6 求本世纪末工农业总产值翻两番所需的平均年增长率。

```
10  LET I = 0
20  LET I = I + 1
30  LET A = (1 + I/100) ↑ 20
40  IF A < 4 THEN 20
50  PRINT I
60  END
```

例7 将输入的三个数从大到小顺序打印出来。

```
10  INPUT A, B, C
20  IF A>=B THEN 60
30  LET W=A
40  LET A=B
50  LET B=W
60  IF C<B THEN 100
70  IF C>A THEN 150
80  PRINT A, C, B
90  GOTO 200
100 PRINT A, B, C
110 GOTO 200
150 PRINT C, A, B
200 END
```

例8 挑出全班学生中的最高分与最低分。

```
10  INPUT A
20  LET H=A
30  LET L=A
40  INPUT A
50  IF A>200 THEN 200
60  IF A>H THEN 90
70  IF A<L THEN 120
80  GOTO 150
90  LET H=A
100 GOTO 150
120 LET L=A
150 GOTO 40
200 PRINT H, L
```

```
210  END
```

例9 把40名同学的成绩按从高到低的顺序排列起来并输出。

```
 10  DIM A (40)
 20  FOR I = 1 TO 40
 30  INPUT A (I)
 40  NEXT I
100  FOR I = 1 TO 39
110  FOR J = I + 1 TO 40
120  IF A (I) >= A (J) THEN 160
130  LET W = A (I)
140  LET A (I) = A (J)
150  LET A (J) = W
160  NEXT J
170  NEXT I
200  FOR I = 1 TO 40
210  PRINT A (I)
220  NEXT I
250  END
```

第六章 外部设备的使用

目前使用的普及型计算机，大多是采用8位的微处理器组成的微型计算机，也就是人们常说的8位微型机。所谓8位机，就是微处理器内部寄存数据的寄存器和进行数据处理的运算器都是八位的，能同时寄存或处理8位二进制数字。在微处理器外部，有八条数据线，同时传送八位二进制数字，它还有十六条地址线，可以不重迭地选择$2^{16}=65536$个即64K个存贮单元。因此一般8位机的内存贮器是64K字节。当然许多普及计算机不一定需要64K内存，可以只做几K到几十K字节内存。

前面已经讲过，微型计算机的主机包括微处理器（也称中央处理单元CPU）、内存贮器（64K或不足64K字节）和输入/输出接口三个主要部分。普及型计算机一般都将键盘和主机板做在一起，我们称之为带键盘的主机。显示器、印字机、数据记录器、游戏操纵杆、大存贮量的磁盘驱动器等等都称为外部设备。这一章介绍普及型计算机中常用的几种外部设备。

6-1 显示器

在专用的微型机系统中，都采用专门用作显示的监视器。

这种监视器屏幕显示的图形、字符等一般为绿色，也有彩色的。它不能接收和显示广播电视信号，只能显示从计算机电缆送来的字符和图形。不过监视器的清晰度都比一般电视机高，其显示的图象一般比较稳定和清晰。普及型计算机为了使用方便和减少费用，大多选用家用电视机作为显示器。使用黑白电视机就可以进行BASIC语言程序练习，还可以做出各种图案。有条件为它配上彩色电视机作为显示器，更可以把计算机的色彩功能发挥出来。

显示器与计算机的连接方法已在第一章里介绍过，这里就不再重复了。

普及型计算机输出给电视机的信号频率的频道，随计算机型号的不同而有所不同，你拿到计算机时一定要问明白，或者查阅手册。

6-2 数据记录器

普及型计算机的内存贮器最大为64K字节，而且用户存入内存的程序在关闭电源后就会丢失。为了扩大内存容量和长久保存信息，计算机一般都配有外存贮设备。数据记录器就是最简易的一种外存设备，普及型计算机一般配有自己型号的数据记录器，也可采用通用的磁带录音机。

要连接录音机，可使用所附的一对专用电缆。这对电缆每一条两头各有一个3.5mm的双芯插头。其中一条是用于将计算机的输出信号送入录音机。它一头插计算机CASSETTE OUT或MIC插孔（即信号输出插孔），一头接录音机MIC或MICROPHONE插孔（即话筒插孔）。电缆中的另一条是用于将录音机的信号送回计算机，它一头插录音机标有EAR或EA

RPHONE或MON或MOMTOR的插孔，即外接耳机插孔；另一头插计算机CASSETTE IN或EAR插孔。将录音机和计算机接通电源，就可应用了。

数据记录器录存计算机程序和数据，以及放送程序或数据给计算机，都是在计算机的命令控制下进行的，操纵者在键盘上按下了相应命令（屏幕上也会有显示）计算机就会协调地完成上述动作。各种不同型号的普及计算机关于使用盒式录音机贮存程序和数据的命令都有所不同，这里我们举一种型号COMX-35微型机的例子，为你使用录音机存放程序和数据提供一些帮助。

在COMX-35微型计算机中有四条用于录存程序和数据的命令。PSAVE、DSAVE用来将程序和数据录入磁带；PLOAD、DLOAD用来将录在磁带上的程序和数据从录音机取回计算机，下面我们分别介绍这四条命令。

一、PSAVE

此命令是将程序从计算机转录存入录音机。在使用此命令之前，首先应按前面讲过的办法将计算机和录音机用电缆线连好。先用LIST命令检查一下你所要录入的程序，检查无误后可开始进行录入。这时你一定要注意调整一下录音机的录音电平控制旋钮。这个旋钮位置放的不同，有时对录入成功与否影响很大，可多试几次找出规律。如果所用录音机在录音时是用自动电平控制的，则可不管。

1.将录音带倒到起始位置，并将磁带计数器置零。这样你可记住程序的起始位置。如果你的磁带上已存有一部分录好的程序，你可倒过这段程序，同时记下记数器的数值。请记住，你每次录入程序时，要记下每段程序起、止点的计数器指示

数。

2.按下录音机的相应键，使录音机进入录音状态；

3.在计算机键盘打入PSAVE命令，再按CR 键。这就意味着开始将程序录入磁带。这时显示屏上光标消失，同时在录音机扬声器中可听到不规则的叫声。

4.当程序全部录完后，显示器上会重新显示“READY”和光标。这时请你记下录音机计数器上的数码。

在录存过程中，如上所说，一开始是一阵尖叫声，然后是一阵噪声，表示在传送程序的第一页（256位数一组，称为一页）。接着是一阵短促的尖叫声，这是页头信号，标志第二页开始，以后又是噪声，表明在传送程序的第二页。再往后尖叫声和噪声交替出现，一直到整个程序传送完毕。传送结束的标志是，尖叫声之后继以低音。当你使用熟练之后，你就会熟悉上述各种声音出现的顺序，并大致掌握程序已录到什么地方。

二、DSAVE

如果录存的是数据而不是程序，则可用此命令。

用DSAVE命令录入数据的步骤，与上面讲PSAVE 时的一样，只是将PSAVE命令改为DSAVE命令就行了。在 录入的过程中，同样也有存程序时产生的尖叫声和噪声。

三、PLOAD

PLOAD命令是指将程序从录音机取回计算机

1.将录音机与计算机联接好。将装有程序的录音带装入录音机，并找到程序开始的地方。将音量开关调至适当位置（通常先放在中间位置试试）。

2.这时要弄清计算机中现在存放的程序或数据是否已经不

再需要了，因为下一步调入磁带中的程序时会清除内存贮器的全部内容，原来存在存贮器中的程序或数据将会全部洗掉。

3.如果计算机显示出“READY”和“：◆”符号，你可打入PLOAD命令，然后按下录音机的放音键，当听到一声尖叫声时，立即按CR键开始接收程序，这时光标消失，并伴有尖叫声和噪声。

4.当程序全部取入计算机之后，显示器上重新显现“READY”和“：◆”符号。

四、DLOAD

是将数据（不是程序）由录音机取出装入计算机的命令。它是将先前由DSAVE命令存入磁带上的数据取回计算机，其操作步骤与PLOAD相同，只需将PLOAD命令改为 DLOAD 命令即可。

有了磁带数据记录器，你就可以把一些需要长久保存的程序或数据存在磁带上。如果你今天编写的程序没有完全调试好，或者有些部分还要增删，你都可以将已经输入计算机的内容存入磁带。明天当你再继续调试程序时，你就可以用前面讲的办法，将磁带的内容调入计算机，从而免去了你重新一个字符一个字符敲键盘的工作，提高你的工作效率。

使用数据记录器往往不会一次成功，这需要积累经验，你应当耐心地多实践几次，掌握要领后（主要是音量调节，并记住程序在磁带上的位置），就会应用自如了。另外，存放时所用的录音机和取出时用的录音机一定要用同一部录音机，否则不易成功。

6-3 印字机

一般标准的8位微型机都配有一台打印机，用来将程序、数据、图表等有用内容打印在纸上，作为永久的记录。这在计算机中称为“硬拷贝”。普及型计算机的打印设备是印字机，它的结构和打印机不同，每行打印出的字符个数也少，因此我们称之为“印字机”。各种不同型号普及计算机都配有自己的印字机，而且各不相同。控制印字机动作的命令也不一样，这一节我们介绍一种典型的印字机及其简单操作。

印字机和带键盘主机之间，除带状电缆以外，还需加一个印字机接口部件，这样才能将它们连接起来。印字机在主机的控制下，进行印字操作。主机和印字机的连接方式如图6·3·1所示。

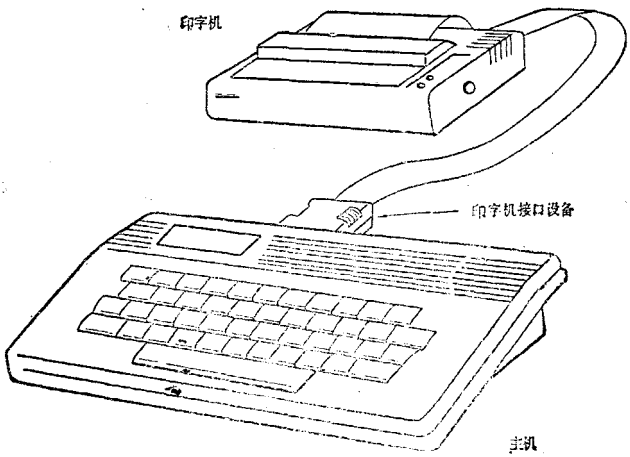


图 6·3·1 印字机和主机的连接

要让印字机进行打印字符或图形的动作，必须由计算机发出控制命令，即由操作者用键盘输入命令。这一节我们以LASER-310型计算机为例，简要介绍一下印字机的使用。LASER-310机有关印字机的命令有三个：LLIST，LPRINT 和COPY。前两个命令和我们学过的列表LIST和打印PRINT命令语句相似，COPY是“拷贝”命令。

LLIST

这个命令和屏幕LIST命令功能相似。LIST将计算机内的程序列在显示屏上，而LLIST则是把这些内容送给印字机，由印字机在印字纸上印出列表的内容。

LPRINT

这个命令和PRINT 功能相似。PRINT命令将其语句中要打印的内容在屏幕上显示出来，而LPRINT命令则把要打印的内容由印字机印在打字纸上。

COPY

这个命令是将显示器屏幕上显示的所有内容打印出来，好象给显示屏“复制”似的。你可以将所要打印的内容先“写”在显示屏上，然后键入COPY命令，计算机运行后，印字机就不断印出显示器屏幕上的内容。

当你的印字机正在印字时，如果你想停止打印，你可以按动键盘上的BREAK键（CTRL_-）这时印字机就会停下来。

普及型计算机所用的各种印字机，功能相仿，型号很多，但多不能通用。因此当你为计算机选配印字机时，注意配接相应的印字机。

第七章　普及型微计算机的特殊功能

前面已经介绍了微计算机的一些功能，可是，除此之外，它还有不少特殊功能，现在就来介绍其中常用的几种。

7-1　屏幕作图

在显示屏幕上除了显示字符以外，很多计算机还有绘图的能力，在第一章中你已经看到一些例子了。

为什么微型计算机能有屏幕作图功能呢？为了说明这个问题，我们先来看看在屏幕上是如何显示字符的？它是按照一定的行、列格式显示的，比如有的机器就是按24行、每行40列显示的，换句话说就是把整个屏幕分成40×24＝960个小格子。你再细看一看就会发现，在每个格子范围内又是由许多小点点组成，比如由7×8＝56个点组成。这56个小点有的亮、有的不亮，就可以显出“P”、“3”、“=”等等不同的字符形状来。既然如此，用这56个点当然也可以组成其它的图形，比如矩形、三角形、菱形等等。此外，从整个屏幕来看，则是由280×192个点点组成，只要加以控制，使其中有的点亮、有的

点不亮，完全可以组成更复杂一些的图形。所以计算机的屏幕绘图功能通常有两种方式。第一种方式是以原来那些小格子为范围，在格子内组成一些图形，再把一个个格子连成一片，可以拼成大图形。第二种方式就是不拘泥于原来的格子，直接在整个屏幕上（例如280×192=53760个点点）用小点点绘图。这样，虽然麻烦一些，但是可以画出细致的图形。在不同的机器上，实现屏幕绘图的具体操作各不相同，下面我们举几个例子。

COMX—35计算机只有上述第一种方式的绘图功能。但它允许使用者在一个小格子的范围之内自己任意规定一种形状。

当机器开机时本身内部就带有一部分已建立好的图形符号，分别对应着从0～255的代码。其中一部分就标在键盘上，按相应的键就可以调用；另一部分则只能直接使用代码（用CHR$语句就可以把这些代码转换成符号）。为了在屏幕上显示出这些图形，应当用PRINT语句。比如我们执行一下这个程序：

```
10  FOR I=0 TO127
20  PRINT I, CHR$(I)
30  NEXT I
40  END
```

就会显示出一系列符号来，包括字母、数字、标点、图形和空格。再比如，“R”键右下角画着一个乐符，你打一下PRINT[SHIFT—R]（按下SHIFT键同时再按R键）就会显出这个乐符来。由于“SHIFT—R”的代码是114，所以打PRINT CHR$(114)也能得到同样的结果。把几个符号连起来也是很有意思的，例如你试着打一下：PRINT CHR$(111,112,

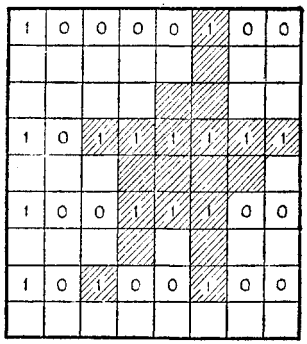
112，112，113)结果，5个图形连起来，显示出一架大飞机。

除了这种机器内部规定好了的图形之外，COMX-35还允许使用者自己重新定义某个代码所对应的图形符号。比如，刚才讲到，“SHIFT—R”的代码是114，原来它对应的图形是个乐符。你可以在程序中写上一个SHAPE语句(定形语句)，把它对应的图形重新定义为一颗小星星：

```
10  SHAPE(114, "84848CBF9E9C94A400")
20  PRINT CHR$(114)
30  END
```

定义之后再用PRINT语句显示CHR$(114)或是“SHIFT—R”，屏幕上显示的就不再是乐符而是一颗绿色的小星星了。直到重新开机（或按RESET）之后，114代码对应的图形才又恢复为乐符。SHAPE语句的格式如上例所示，括弧中第一部分是你指定要使用的代码（0～255中之一个）；第二部分在引号内是十八个16进制数。它是用来描画图形符号的。我们举个例子来说明如何写成的。比如上面提到的画一颗小星星，你首先在纸上画出九行八列小方格（见图7·1·1），右边六列（共54个）小方格用来画上你想要画的图形，——并把图形中的面积打上阴影线。然后把凡是画上阴影线的方格内填上1；未画上的方格内填上0。最左边两列从上到下可以填九个两位数。它们代表你所希望该行显示的颜色：00代表黑，01代表蓝，10代表绿，11代表浅蓝。这样填好之后，就得出每行有8个二进制数，将它分成左、右两个四位数码：

```
1000 (8) 0100 (4)
1000 (8) 0100 (4)
1000 (8) 1100 (C)
1011 (B) 1111 (F)
```



（原图6×6cm）

图 7·1·1 自定义图形示例

1001（9）1110（E）
1001（9）1100（C）
1001（9）0100（4）
1010（A）0100（4）
0000（0）0000（0）

每个四位码可以转换成一个十六进制数（如上面括弧中所写的那样），于是从左到右、从上到下就可从写出十八个十六进制数来：

84848CBF9E9C94A400

这就是应当填入SHAPE语句中的值。

Apple-II plus计算机具有前面讲到的两种绘图方式。不过第一种方式比较简单，没有很多的图形符号，只是一种——由 7 × 4 点组成小矩形亮点。这叫低分辨率作图。用一个个小矩形就可以在屏幕上组成较粗的各种图案来。在程序中进行低分辨率作图，首先要用一个 GR 语句（绘图语句），然后分别用以下几个语句：

例 PLOT15, 23

它的意义是把第16列、第24行那个位置上的小矩形画出来；

（注意：整个屏幕共有48行40列个矩形，行号由 0 到47，列号由 0 到39，所以第16列的列号为15。）

HLIN 5, 34AT11

它可以画一条水平线（也就是把一系列小矩形水平地排起

来）由第 6 列画到第35列（从左到右），上下位 置在 第12 行处。这里HLIN表示水平线条之意。

VLIN 3, 36AT14

它可以画一条垂直线（也就是把一系列小矩形垂直地排起来），由第 4 行画到第37行（从上到下），左右位置在第15列处。VLIN表示垂直线条。

下面这个程序可以用来画出一个边框，并从左上角到右下角画一条对角线（其中15语句的含义下面会讲到）。

```
10  GR
15  COLOR = 3
20  HLIN0,39AT 0
30  VLIN1,38AT 0
40  HLIN0,39AT39
50  VLIN1,38AT39
60  FORI = 1TO38
70  PLOT I,I
80  NEXT I
90  END
```

Apple-II plus计算机提供的第二种绘图方式叫高分辨率作图。这是直接以整个屏幕的192行、280列个点点画图。下面这个例子是一个画出正弦曲线的程序；

```
10  HGR
20  HCOLOR = 5
30  HPLOT 5,80 TO 270,80
40  HPLOT5,80
50  FORI = 1 TO260
```

```
60  HPLOT TO I,80-75*SIN( I*3.1415/65 )
70  NEXT I
80  END
```

要进行高分辨率作图，首先要用一个HGR语句。其意思就是高分辨率作图。行号为20的语句的含义，是指定图形色别，将在下一节再讲。30行语句的意思是从第81行第6列那一点起到第81行第271列那一点止画一条直线。40语句是只画一点。60语句是指由上次画的末点起到某行某列那一点止画线。

其它一些普及型微计算机的屏幕绘图操作方法也都与上述两种机器类似，只是具体的语句、命令可能略有不同。比如LASER—310机的低分辨率绘图是有16个内部固定的图象符号组成。在程序中应用要首先写一个MODE（0）语句，再调用显示这些图象的符号。它的高分辨率绘图是将整个屏幕分为128×64个点子。首先用MODE（1)语句，然后用SET(n_1,n_2)语句在n_1列n_2行处画一点（$0 \leqslant n_1 \leqslant 127$，$0 \leqslant n_2 \leqslant 63$）。你要想画线条就只有一点一点地去画（可以用循环语句）。还有一个语句：RESET(n_1,n_2)是用来把n_1列n_2行处的原先“画上”的一点再抹去。可见使用起来也是挺灵活的。

7-2 选择色彩

如果你采用彩色电视机或彩色监视器作为计算机的输出显示装置，根据你送入的命令、程序，由计算机主机发出对电视机的控制信号，不但能控制显示的符号形状、位置、还能控制显示的颜色。选择的范围与操作方法因机而异，下面通过几个例子来说明一下。

COMX-35计算机具有较多的色彩选择功能。首先，它可

以用SCREEN语句选择屏幕的颜色，比如在程序中可以有这样一行：

```
50  SCREEN (7)
```

括弧中是一个1～8的正整数，也可以是一个表达式，但这个表达式的值必须是1～8的一个正整数。这个数就表示你要选择什么颜色：

1—黑；2—绿；3—蓝；4—浅蓝；5—红；6—黄；7—洋红；8—白。

下面是一个程序例子，用来使屏幕颜色依次变换：

```
10  FORI=1TO8
20  SCREEN (I)
30  FOR J=1TO200
40  NEXT J
50  NEXT I
60  SCREEN(1)
70  END
```

其次，可以用 COLOR 语句选择屏幕上所显示出的字符的颜色（即图形的颜色）。比如在程序中可以有这样一句：

```
30  COLOR (10)
```

括弧中是一个数或一个表达式，其值是1～12之间的一个正整数，表示选择什么颜色。在COMX—35型机中，一个代码在键盘打入时所代表的颜色和由计算机中存贮器中取出时所代表的颜色是不同的，例如10键入时代表红色而由计算机取出时则代表蓝色。下表中每种颜色后面括弧中的数字是发光率的百分数。百分数愈高，颜色就愈亮。

第三个语句是CTONE语句，它可用来调节色调(饱和度)。比如在程序中有这样的语句：

代　码	计算机输出时		键盘打入时	
1	黑	(0)	绿	(59)
2	红	(30)	黄	(89)
3	蓝	(11)	浅蓝	(70)
4	洋红	(41)	白	(100)
5	黑	(0)	蓝	(11)
6	红	(30)	洋红	(41)
7	绿	(59)	浅蓝	(70)
8	黄	(89)	白	(100)
9	黑	(0)	红	(30)
10	蓝	(11)	洋红	(41)
11	绿	(59)	黄	(89)
12	浅蓝	(70)	白	(100)

```
20  CTONE (2)
    ⋮
    ⋮
100  CTONE (0)
```

CTONE后面括弧中的数如果取非零值，就可以在颜色不变的情况下使显示字符的色调加深。如果取零值，就取消上述效果。

综合运用上述三种语句，那么鲜艳的色彩效果就出来了。不过应注意几点：

①开机后正常情况下是没有色彩效果的，相当于SCREEN（1）和COLOR（2）的情况。直到执行了有关命令后，才会按上述规定改变颜色。而且这以后颜色又保持不变，直至有新的命令。

②屏幕颜色和显示字符的颜色当然不能选成一样（那样就看不出字符了），应当在色彩上搭配适当，还应根据发光率来

搭配以使对比明显。

③我们知道，电视机显示的色彩是由输入的某些电信号和机内电路来控制的，其中影响因素较多。所以实际得到的色彩很可能与上面列的表不完全一致，这是正常的。建议你不要拘于上面给出的"代码——色彩"对照表，最好还是在你自己的机器系统上做做实验，获得确切结果作为以后使用的依据。

④COMX—35计算机有个特点，键盘打入时和由计算机中调出时的显示颜色的代码不同颜色不一样，这样便于你察看。使用时应结合屏幕颜色一起考虑、调配适当。

LASER—310计算机无论在MODE（0）或MODE（1）哪种显示方式下，都可以用COLOR语句选择颜色，语句格式为：

```
COLOR A, B
```

其中，A为1～8范围的正整数，用来选择显示字符的颜色；B取0或1，对应于两种屏幕底色。你可以运行一下这个程序：

```
10 MODE (1)
20 FOR I=0 TO 1
30 FOR J=1 TO 8
40 COLOR J, I
50 FORK=63-J*5TO63+J*5
60 SET(K, 31-J*3)
70 SET(K, 31+J*3)
80 SET(63-J*5, K/2)
90 SET(63+J*5, K/2)
100 NEXT K
110 FORN=1 TO200
```

```
120 NEXT E
130 NEXT J
140 NEXT I
150 END
```

它可以把各种颜色依次表现出来。

Apple-IIplus计算机在低分辨率绘图方式中，画出的一个个小矩形可以有16种颜色（包括黑色和白色），其代码分别为0～15。在程序中用下面这样的语句进行颜色的选择，例如：

```
50  COLOR = 3
80  COLOR = 12等等。
```

在高分辨率绘图方式中，色彩比较贫乏。选择颜色所使用的语句是这种样子：

```
150 HCOLOR = 2
```

颜色代码有8种（0～7）。不过其中0、4和3、7分别为黑色和白色，所以只有其余4种代码有意义。这是在有限的内存贮器容量条件下，牺牲了色彩选择方面的功能，换取来绘图方面的高分辨率。

7-3 音 响

许多普及型微计算机为了满足个人、家庭在娱乐方面的需要，都具备音响功能。在机器内带有一个扬声器。根据命令或程序，由计算机产生具有一定频率和幅度的电信号，推动扬声器，发出各种音响效果。具体实现方法和操作命令因机而异，下面比较详细地介绍一下COMX—35计算机在这方面的功能，你可以把它作为一个例子来理解。以后在其它型号计算机上进

一步摸索时，就不会太陌生了。

COMX-35计算机的音响功能较多，使用灵活简便。它可以发出三种音响效果：音乐声、单音调声和白噪声。

用来产生音乐声的语句为 MUSIC，语句格式在第一章中已介绍过是MUSIC(X,Y,Z)其中X、Y与Z分别表示所需要的频率范围和幅度。X取值范围是1、2、3、……7，分别对应于乐声中的"都(do)、来(ray)、咪 (mi)、……提(ti)"七个音符，即简谱中的"1、2、3、4……7"。Y的取值范围是1、2、3、……8，表示有八个音阶可供选择。音阶又叫"八音度"或"倍频程"，比如简谱中1、2、3、4、5、6、7、i就形成了一个"八音度"，它的上限频率是下限频率的两倍，比如：i的频率是1的两倍，是1的4倍等等。把X和Y的取值配合起来就可以得到高低变化范围很宽的56个音符（相当于钢琴上的56个键）。Y取8、X取7时的频率最高。来试一下：

```
10  FOR I=1 TO 8
20  FOR J=1 TO 7
30  MUSIC(J,I,5)
40  FOR K=1 TO 200
50  NEXT K
60  NEXT J
70  NEXT I
80  MUSIC(0,0,0)
90  END
```

MUSIC语句里括孤中的第三个数是控制音量的，取值范围为0、1、2、……15，对应着音量由弱到强。你看，MUSIC 语句不是很好用吗！任何一个识简谱的人都可以用它使计算机"唱"起歌来的。

当然这里有个问题：如何控制发某个声音的时间长短呢？有些其它型号普及型计算机的音响功能语句中包含有发声时间控制参数，不过COMX-35机并没有，只好采用其它办法。比如上例中的40～50语句是一组循环语句，你可能已经注意到，其中没有循环体，也就是说每次循环当中不必做什么。那么，循环起什么作用呢！这里，它只起一种控制延时的作用，使30语句发出的音响持续一段时间，再转为下一个音符。故K的数值愈多循环次数就愈多，每一音的延时就愈长。当然，这是一种不精确、不直观的控制方法。要想达到预定的延时，循环次数取多少才合适，要靠实验确定。下面再看一个例子，让机器奏一小段曲子：

51 54 | 32 10 | 11 23 | 31 34 | 5—— |

程序为：

```
 10 FOR I=1 TO 2
 20 READ A,B
 30 MUSIC(A,B,4)
 40 FOR J=0 TO 20
 50 NEXT J
 60 NEXT I
 70 FOR I=1 TO 5
 80 READ A
 90 MUSIC(A,4,4)
100 FOR J=0 TO 20
110 NEXT J
120 NEXT I
130 MUSIC(0,0,0,)
140 FOR J=0TO 20
```

```
150  NEXT J
160  MUSIC (1,4,4)
170  FOR J=0 TO 15
180  NEXT J
190  MUSIC (0,0,0)
200  FOR J=0 TO 5
210  NEXT J
220  FOR I=1 TO 7
230  READ A
240  MUSIC(A,4,4)
250  FOR J=0 TO 20
260  NEXT J
270  NEXT I
280  MUSIC(5,4,4)
290  FOR J=0 TO 80
300  NEXT J
310  MUSIC(0,0,0)
320  DATA 5,4,1,5
330  DATA 5,4,3,2,1,1,2,3,3,1,3,4
350  END
```

可见这种控制方法，编写起来比较烦琐。尽管如此，你用MUSIC语句，确实能让计算机“唱”起来的。

但是，仅有MUSIC语句往往不够用，如果你需要半音怎么办？为了模仿生活、自然中的一些音响还需要更丰富的频率变化。为此目的，提供了另一种语句——单音调声

TONE(X,Y,Z)。

其中，Y和Z的含义、取值范围都和在MUSIC语句中的一样，

所不同的是X取值范围为1～128的正整数。也就是说，在一个“八音度”范围内，不再仅仅有七个音符供选择，而是把频率又分为128个“小台阶”，这样就提供了几乎是连续的选择了。

第三种声响是噪音NOISE，语句格式为

NOISE(Y,Z)

其中Y和Z仍然是分别用来选择音调和音量的。这个语句产生一种高斯分布白噪声。什么叫白噪声？当没有节目时你打开电视机、放大音量，听到的就是一种白噪声。它包含一系列的频率成份。在产生音响效果时这种声音常用到。

以上这三种产生不同音响的命令有共同的特点：任何一个命令一旦被执行，就一直发出这个指定的声音，直到又有新的音响命令为止。要想停止发声，则分别用：MUSIC(0,0,0)、TONE(0,0,0)、NOISE(0,0)。控制发声持续时间的方法，除了前面介绍过的用循环语句以外，更好的办法是利用机器本身的时间控制功能，详见下一节。在那里我们将给出更直观的控制延时语句，在那一节末尾，还要再给出一个“机器奏曲”的例子。适当地组合这三种声音命令，就可以合成一支乐曲。由于这些都可以编在程序里，于是可以让音响伴随着图形与色彩的变化，得到生动活泼的输出。

与上述三个语句配合使用的还有一个大范围音量调节语句

VOLUME(X)。

其中X取值为1、2、3、4，相应的音量由弱到强。在程序中应当把这个语句安排在上述三个语句之前，等于先对音量规定一个大范围，然后再由各语句中的Z参数做小范围的调整。二者配合就能使音量有64种选择。

7-4 时间控制功能

一、实时控制

实时控制本来指的是在某物理过程或现象出现的同时进行计算、判断等等处理。因此要求处理的速度必须足够快，使得处理的结果能马上被用来影响或控制正在进行中的这个过程。普及型微计算机的运算速度较低，所配的接口电路、外部设备也简单，一般不能用于工业生产、科学实验过程的实时控制。不过它们通常也有这种及时做出反应的功能。一般有两种情况：一种是对键盘输入及时做出反应；另一种是对“游戏捧”的动作及时做出反应。我们还是以COMX-35计算机作为例子。它的BASIC语言中有一个KEY函数。使用的格式是KEY=某键的代码，在程序中每当遇到这个指令时，记算机就自动地查看一下该代码键是否刚被敲过。通常把它用在一个IF-THEN语句中并不断地循环执行，一查到是某个预定的量就及时做出反应。例如：

```
10  LET I=0
20  LET I=I+1
30  IF I>8 THEN LET I=1
40  SCREEN(I)
50  FOR J=1 TO 50
60  IF KEY=65 THEN GOTO 100
70  NEXT J
80  GOTO 20
100 SCREEN(1)
110 END
```

这个程序使屏幕底色不断地变换，直到按下A键（代码为65）为止。如果把60语句改为：

IF KEY=136 THEN GOTO 100

那就是只有把游戏棒向上扳才能停止。因为136是游戏棒上扳这一动作的代码。COMX-35计算机上有一个简易的游戏棒，它可以向上、下、左、右四个方向扳动，相当于四个按键，代码分别为136、138、139、137。

二、延时控制

利用计算机内的时钟脉冲，可以建立起一种类似于"闹钟"的功能来。就是说，在程序运行过程中，从某时刻开始计时，待达到一个预定时间后就转去执行一个指定的动作。例如，在COMX-35计算机的BASIC语言里有这样两个语句：

WAIT(X)。这是个等待语句它可使程序的运行暂时停止，延迟一段时间以后再接下去继续运行。延迟的时间长短由括孤中的数X来指定，X的值应当是正整数，每1000相当于6.4秒，即每6.4毫秒为1单位。

TIME(X)。计时指令，一个程序在运行过程中一遇到这个语句就开始计时。当计到X时间（X为正整数，每50为1秒钟）就自动调用一个子程序，这个子程序的起始行号应当事先在一个TIMOUT语句中指定。下面这个例子，在第10语句中，指定了这个子程序由200行开始：

```
10  TIMOUT 200
20  TIME(10)
25  CPOS(0,0):CLS
30  CPOS(RND(23),RND(39)):PRINTCHR$
(114)
```

```
 40  IF X=0 THEN GOTO 100
 50  GOTO 30
100  CPOS(0,0):CLS
120  END
200  READ X,Y,Z,T,I,J
210  SCREEN(I):COLOR(J)
220  MUSIC(X,Y,Z)
230  TIME(T)
250  RETURN
300  DATA 5,4,9,50,4,3
310  DATA 3,4,8,50,4,3
320  DATA 1,5,9,50,4,3
330  DATA 6,4,6,50,4,3
340  DATA 5,4,9,75,6,7
350  DATA 4,4,7,25,6,7
360  DATA 3,4,9,100,6,7
370  DATA 5,4,9,75,7,12
380  DATA 4,4,7,25,7,12
390  DATA 2,4,9,100,7,12
400  DATA 4,4,9,75,5,8
410  DATA 3,4,7,25,5,8
420  DATA 1,4,9,100,5,8
430  DATA 0,0,0,100,1,12
```

这个程序是让计算机按一定节奏“唱”段曲子，同时在屏幕上变换着色彩、随机地画出一些“音符”符号来。30～50行就用来反复不断地绘图。与此同时用TIME语句每隔一个确定时间就调用一次子程序。子程序中，一方面控制发声和颜色，另一

方面又用TIME语句确定下次“转子程序”的时间，也就是本次发声的持续时间。

7-5 屏幕编辑(EDIT)

计算机的输出显示装置可以是电视机或专门的监视器。运算结果和通过键盘输入的信息都显示在屏幕上，便于你核对检查已存入的程序和数据。在使用过程中可能要改进或修订，一般的做法是先把它们显示到屏幕上再对照着进行修改。这种对屏幕上显示的信息进行删除、变更、插入等修改工作，就叫做“屏幕编辑”。在使用计算机的过程中，这是很重要很经常的一件工作。我们举个例子，假如原先存入过这样一行程序：

```
230 PRINT A,C,H,I
```

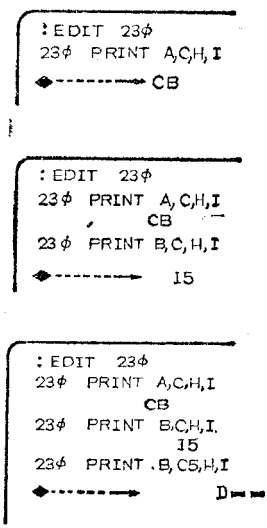


图 7·5·1 屏幕编辑示例

现在要修改一下：把A改为B，把C改为I_5，把 I 删掉。下面先来看看在COMX-35 型计算机上应该如何操作。具体做法是（参见图7·5·1）：

①打命令EDIT 230；

②屏幕下面显示出230行内容，光标在其下一行；

③用空格键把光标移到所改位置，如图 7·5·1 那样分三次分别在要改的位置上打“CB”、“I5”和“D␣␣”（␣表示空格），每次做完后都要打“CNTL-S”，回到第②步；

④全部做完后，再打一次“CNTL-S”。

为什么这样操作就能达到修改目的呢？

你在键盘上敲一个字、屏幕上显出这个字，这个过程在机器内部也是很复杂的，要通过一系列硬件、软件才能实现。一般使用者无需深究。不过为了便于理解“屏幕编辑”功能，掌握并灵活地运用这种功能，我们不妨对有关原理做一些简单、形象（当然不太严格）的描述。下面这一大段讲的是“原理”，如果你一时看不懂，也没关系。继续看下去。等你明白了“怎样做”之后，回过头来再看一遍“原理”，两相对照，就容易理解了。

在计算机中，每个数字、字母、符号都用一串二进制数码来代表（通常采用ASCII代码，见附录）。在内存贮器中有一部分称作“显示暂存区”。它与整个显示屏幕相对应：假若整个屏幕分为24行，每行可以显示40个字符，也就是说屏幕划分为960个位置，那末“显示暂存区”也就包括960个存贮单元，与之一一对应。每个单元中存放着一个二进制数，表示着（更确切地说是决定着）屏幕的那个相应位置上显示的是一个什么样的字符（注意：“空白”也是一种符号）。至于怎样由二进制代码产生出必要的电信号，如何在屏幕上形成相应的字符，这些我们都不去深究。只要明瞭，你想要在屏幕某个位置上显示什么字符，就得在“显示暂存区”的对应单元中存放上相应的二进制码。当你打入一个程序时，并不是立刻作为“程序”存入内存贮器。内存贮器中还有一部分叫做“键盘输入暂存区”，你打进去的每个语句先暂时放在这里，等到你打一下“回车键”（CR或ENTER、RETURN）时才作进一步的处理。每当在键盘上敲一个键，相应的二进制码被送入“键盘输入暂存区”（以下简称“输入区”），同时也被送入“显示

暂存区”（以下简称“显示区”）。连续击键，就顺序送入一系列二进制码。为了在这两个暂存区中自动地确定该存入哪个单元。我们可以想象有两个联动的标记，好像一对“指针”，一个总是指向“输入区”中下一个该存入的单元；另一个总是指向“显示区”中下一个该存入的单元，而且在其对应的屏幕位置上显示一个“光标”。“指针”移动一个单元，屏幕上“光标”相应地移动一格。这样每敲一键，一个二进制码就进入两个暂存区，分别存入各自“指针”所指的单元里（于是屏幕上也就有对应的显示）。然后两个“指针”顺序向后移一个单元。此外，还有两个办法能使这一对“指针”一起移动。一个是敲退格键（有的键盘上标为“backspace”，有的标为“←”有的标为“RUBOUT”，等等），每敲一次此键，两个“指针”都往回移动一个单元，（屏幕上“光标”左移一格）。但是“显示区”中刚退出来的那个单元里原来的代码仍保留着。如果要“指针”往前走，就按恢复键（有的键盘上标为“→”）每按一次此键，在两个“指针”当前所指的两个单元之间就传送一次——把“显示区”里的二进制码送入“输入区”，然后两个“指针”都往前移一个单元（屏幕上“光标”右移一格）。到目前为止，暂存区中存放的二进制码还不能被机器做进一步处理。只有当按“回车键”之后，机器才检查“输入区”，察看从开头到“指针”当前位置为止的内容，区分不同情况，加以处理：①如果是程序，送入内存贮器的“程序区”，按其行号安排；②如果是数据（比如回答INPUT语句而打入的常量值），送入内存贮器的数据区；③如果是正确的命令（比如LIST），就执行；④如果是错误的命令，就显示一个“出错通知”，拒绝执行……等等。然后两个“指针”都要移动：“显示区”的“指针”指向屏幕下一行开头位置所对应的

单元；“输入区”的“指针”则回到该区第一个单元，

初步瞭解了上述“原理”，再来看“怎样做”的问题，你就能够理解，在按“回车键”之前与之后，应当采用不同的修改手段。

在未打“回车键”之前，若想修改刚打入的字符，可以敲“退格键”，使“光标”退到想要修改的地方，再重新敲入正确的字符。

在打了“回车键”之后，若想修改以前打入的某些程序行，就不能那末方便了。因为它们已经被存入内存贮器的“程序区”了。要想修改某一行，可以重新打入该行的行号和新的语句，然后敲“回车键”，就把这新行也存入“程序区”。由于在“程序区”中是按行号存放，所以对于同号的两行，是用后打入的一行“挤走”原先的该行的内容。这样就达到了修改的目的。你再想一想，如果原先的程序中有一行的行号为80，现在再重打一个行号80，然后紧接着打“回车键”，会是什么结果？——这是用空行取代原有行，也就是把原有行全部删除了。

显然，这种修改方法比较麻烦，特别是当要修改的那一行程序比较长、比较复杂时，也很容易再发生错误。所以，计算机通常又提供一种“屏幕编辑”功能，便于你进行修改。各种机器上具体做法略有不同，都是利用两个暂存区及其彼此的联系，象刚才举的例子中所使用的COMX-35计算机，是用EDIT命令启用“编辑”功能。它把要修改的那一行从“程序区”取出又放入“显示区”和“输入区”，然后只移动“显示区”“指针”——把“光标”移到屏幕的下一行开头。用“空格”键把“光标”移到所需位置（“输入区”的“指针”同时移动），再打“I”、“D”或“C”键分别表示这后面的动作是插

入（I）、删除（D）或修改（C），（只能选择其中一个）。后面跟着打入所需的字符。做完之后打“CNTL-S”表示完成一次编辑。再打一次“CNTL-S”表示退出编辑功能。这一行程序重新存入“程序区”。

对Apple-Ⅱplus计算机，启用编辑功能的命令是“ESC”。它把两个暂存区“指针”之间那种联动关系取消，“输入区”的“指针”保持不动，用I、M、J、K四个键分别向上、下、左、右移动光标（也就是说只移动“显示区”的“指针”）。到达需要的位置后再按除I、J、M、K以外的任何一个键，取消编辑功能（恢复“指针”间联动关系）。再输入所需的字符。所以对于上面那个例题，具体做法是：

①打LIST 230 命令；

②屏幕上显示出原来的230程序行；

③重打行号230，然后按“ESC”键，再反复用“I”键把光标移到上面显示的230程序行的始端（“P”字符之前）；

④反复用“→”键把光标右移到字符“A”处（此时，两个“指针”是联动的，依次把“显示区”中的P、R、I、N、T“空格”一一存入了“输入区”）；

⑤依次按“B”、“→”、“→”键，把“B”、“，”、“C”存入“输入区”，此时光标停在C后面的逗号处；

⑥按“ESC”键、“M”键，把光标下移一格，再连按两次“5”键（注意，第一次是取消编辑功能，第二次是输入一个“5”）；

⑦按“ESC”、“I”、“J”键，把光标移回到原来第⑤步结束时的位置；

⑧连续按“→”键，使光标移到H后面的逗号处；

⑨按“回车键”，完成修改。

你可以再用LIST 230命令检查一遍修改结果。

到现在为止，这一章向大家介绍了计算机的一些特殊功能。再强调一遍，各种计算机上所实现的特殊功能和操作方法是各不相同，甚至差异很大的。这里，我们仅仅是通过一些实例做些入门介绍，帮助你从认识“个别”走向认识“一般”，明白了一般道理，再回到你那台计算机上去，指导你去实践。

第八章　程序设计实例

你已经快读完这本书了，可能心中还有些嘀咕："我真的能使用计算机解决实际问题了吗？"我回答是肯定的。如果你确实边看边练，明白了书中所讲的内容，那么你就的确掌握了必要的使用微机的基本知识。虽然还很初步，但你确实是跨进了"计算机应用"的大门。为了帮助你逐步掌握怎样在实际问题中应用计算机，下面再介绍几个例子，你可以在自己的计算机上试一下。一般说，解决一个问题的途径不止一个，例中所编写的程序并不是唯一的，也不一定是最好的。大家可以举一反三，扩展思路，逐步达到熟练地编写程序。

例1　求解一元二次方程

$$ax^2+bx+c=0$$

从代数中知道，一元二次方程求根的通用式为：

当$a\neq 0$，$D=b^2-4ac>0$时，有两个相异实根：

$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

当$a\neq 0$，$D<0$时，有一对共轭复数根：

$$x_{1,2} = -\frac{b}{2a} \pm \frac{\sqrt{4ac-b^2}}{2a}i$$

当$a \neq 0, D = 0$时，有两个相等实根：

$$x_1 = x_2 = -\frac{b}{2a}$$

当 $a = 0$，$b \neq 0$ 时，只有一个根：$x = -\frac{c}{b}$

当 $a = 0$，$b = 0$ 时，方程无意义。

根据这样的解法，可以画出流程图（见图8.1）。由于要区分不同情况采用不同的计算公式，应当采用“分支”式的流程。流程图中某些框的外面标的数字是对应的程序中的行号。

根据流程图写出的程序如下：

```
10  PRINT "THE SOLUTION OF THE QUADR-
    ATIC EQUATION"
20  PRINT "A*X*X+B*X+C=0"
25  PRINT
30  PRINT "A,B,C=";
35  INPUT A,B,C
40  PRINT
45  LET D=B*B-4*A*C
50  IF A<>0 THEN GOTO 200
60  IF B <>0 THEN GOTO 150
70  PRINT"THE EQUATION IS MEANINGLESS"
80  GOTO 400
150 LET X=-C/B
160 PRINT "SINGLE ROOT X=", X
170 GOTO 400
```

```
200  IF D>0 THEN GOTO 250
220  IF D<0 THEN GOTO 290
230  PRINT "2 COMMON ROOTS X1=X2=";
     -B/A/2
240  GOTO 400
250  LET  X1=(-B+SQR(D))/2/A
260  LET X2=(-B-SQR(D))/2/A
270  PRINT "2REAL ROOTS:", "X1="; X1
275  PRINT, "X2=", X2
```

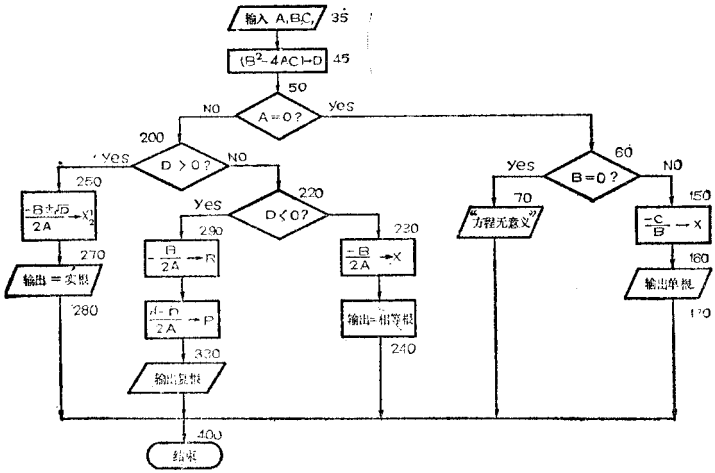


图 8·1 求解一元二次方程的流程图

```
280  GOTO 400
290  LET  R = - B/2/A
300  LET  P=SQR(-D)/2/A
310  IF P>0 THEN GOTO 330
320  LET  P = - P
```

```
330  PRINT "2CONJUGATE COMPLEX ROOTS;"
340  PRINT "X1=";  R;  "+ I(";  P;  ")"
350  PRINT "X2=";  R;  "- I(";  P;  ")"
400  END
```

例 2　我们来看一个物理问题：有一个光滑斜面与水平桌面成α角，设有一质点在 t = 0时由此斜面顶点 A 处开始从静止状态自由释放。如果忽略摩擦力，斜面长度 S = 300cm，α = 65°，求 t = 0.1，0.2，0.3，……1.0秒时质点的速度。

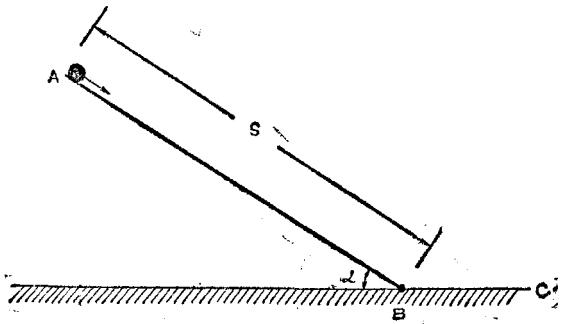


图 8·2　质点在斜面上的运动

由力学知识可知：质点在斜面上运动时，作用在它上面、平行于斜面的力为m·g·sinα，加速度为a = g·sinα。当在水平面BC上运动时，速度为常数，保持它在B点时的速度（作匀速运动）。由A点到B点间的速度V按下式计算：V = at = g·(sinα).t从A到B所用的时间为t_B，$t_B=\sqrt{\frac{2S}{a}}=\sqrt{\frac{2S}{g\cdot\sin\alpha}}$，于是到B点时的速度为$V_B = a\cdot t_B = \sqrt{2aS} = \sqrt{2\cdot S\cdot g\cdot\sin\alpha}$。经过这样一分析，思路就清楚了。比如，可以先计算出V_B，然后按V = gtsinα公式计算 t = 0.1、0.2、……1.0秒时质点速度，最后

判断一下，凡速度大于V_B者都改为和V_B一样大。于是按这个思路画出流程图（如图8.3），并据此写出BASIC源程序：

```
10  LET  R = 65 * 3.14159/180
20  LET  S = 300
30  LET  A = 980 * SIN(R)
40  LET  V1 = SQR(2 * A * S)
50  FOR  T = 1  TO  10
60  LET  V = A * T/10
70  IF  V>V1  THEN LET
        V = V1
80  PRINT"T = " ;  T/10;
        "SEC" , "V = " ; V;
        "CM/S"
100 NEXT  T
120 END
```

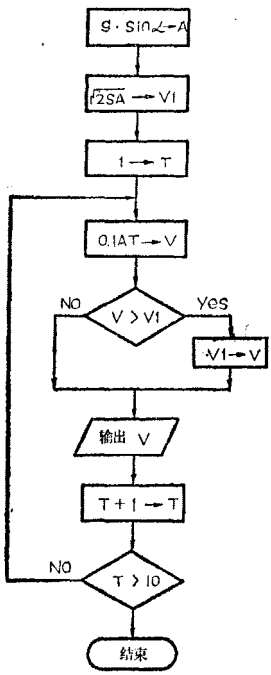


图 8·3　计算斜面上质点速度的流程图

你能不能把这个程序改一下，使它能适应于不同的S和α的值？

从这两个例子我们还可以看出，“神奇的电脑”也并非“天生”来就会解决任何实际问题的，其实都还是靠你自己、即靠“人”教给它才行。具体来说，你拿到一个实际问题，应当首先弄清它的含义，依据有关的专业知识找出其内在的关系和规律，列出公式，再根据数学知识找出这些式子的解算方法，可以画出流程图用以帮助你从

事逻辑思维和判断，然后根据算法再编制程序。

画出流程图和写出BASIC源程序之后，都应仔细检查、改正错误。最后还要把源程序送入计算机进行试算。如果发现错误，还要返回头检查，看看是流程错，还是源程序写错了？都修改好了才能正式使用。

当然，上面讲的这些都属于一般过程，当你要解决的问题很简单、规模很小，那么上述步骤中有些就可以省略或合并。

例3 一个用角度制表示为A°（度）B′（分）C″（秒）的角，用弧度制表示则为D（弧度）。编写一个BASIC程序进行“角度化弧度”、“弧度化角度”。

角度化弧度的计算公式是：

$$D=\frac{\pi}{180}\left(A+\frac{B}{60}+\frac{C}{3600}\right)$$

弧度化角度的计算公式是：

$$A=\frac{180}{\pi}D \quad \text{舍去小数部分取整，}$$

$$B=\left(\frac{180}{\pi}D-A\right)\times 60 \text{舍去小数部分取整，}$$

$$C=\left[\left(\frac{180}{\pi}D-A\right)\times 60-B\right]\times 60$$

四舍五入取整。

下面程序中，借用键盘符号“＊”，“，”“,,”表示度分秒，例如27°41′34″将显示为27＊41，34,,的形式。角度化弧度的程序为：

```
10  REM CONVERT DEGREE TO RADIAN
20  PRINT "A*B, C,,=";
30  INPUT A, B, C
```

```
40  PRINT
50  LET D=3.14159*(A+B/60+C/3600)/180
60  PRINT A; "*" B; ", "; C; ",,="; D
70  END
```

（第10行注释语句REM后面的一串英文字是指“将度化为弧度”之意，也可用汉语拼音打出）

弧度化角度的程序为：

```
10  REM CONVERT RADIAN TO DEGREE
20  PRINT "D(RADIAN)="
30  INPUT D
40  PRINT
50  LET D1=180*D/3.14159
60  LET A=INT(D1)
70  LET B1=(D1-A)*60
80  LET B=INT(B1)
90  LET C=(B1-B)*60
100 LET C=INT(C+0.5)
110 PRINT D; " (RADIAN)="; A; "*"; B;
    ", "; C; ",,"
120 END
```

（第10行语句是英文：“弧度化为度”，对小学生，最好改用汉语拼音）

例4 编写一个“小学生个位数加法自我测验程序”。运行这个程序，计算机开始出题，并等着你回答。（在键盘上打出答案数字）。如果你的答案正确，计算机便显示：“IT IS RIGHT。”并问你“WOULD YOU GO ON？”，如果你愿意继续做下去，就回答YES。那么计算机就又出一道题。如

果你答错了，计算机会显示："IT IS WRONG"。"DO IT AGAIN。"允许你有两次改正的机会。到第三次仍答不对，就又出新题目了。当你不愿意再继续做下去时，计算机会把你总共做了多少题，其中做对了多少、原先做错后又改正的有多少，一直没有做对的有多少，等等，都统计出来。

```
10  REM EXAMINATION
20  LET  I = 0
30  LET  R = 0
40  LET  W = 0
50  LET  W2 = 0
60  LET  I = I + 1
70  LET  J = 0
80  LET  A = INT(10*RND(1))*
90  LET  B = RND(9)
100 PRINT A; "+"; B; "=";
110 INPUT C
120 IF A + B = C THEN GOTO 200
130 LET  J = J + 1
140 PRINT "IT IS WRONG."
150 IF J<3 THEN GOTO 180
160 LET  W = W + 1
170 GOTO 60
180 PRINT "DO IT AGAIN."
190 GOTO 100
200 PRINT "IT IS RIGHT."
```

---

* Apple Ⅱ机上，这里应写INT(10*RND(1))，在COMX-35机上，RND函数后面不必写参数。

```
210 IF J=0 THEN GOTO 240
220 LET W2=W2+1
230 GOTO 260
240 LET R=R+1
260 PRINT "WOULD YOU GO ON";
270 INPUT A$
280 IF A$="YES" THEN GOTO 60
290 PRINT "NUMBER OF QUESTION:";I
300 PRINT "RIGHT:";R
310 PRINT "CORRECT:";W2
320 PRINT "WRONG:";W
330 PRINT "GOOD-BY."
340 END
```

例5 写一个程序，使它能分解出某个数的因子

被分解的数由INPUT语句输入给变量N。

N先被2除，每除尽一次就打印一个“*2”，直到不能再被2除尽。然后再把商用3除，每除尽一次打印一个“*3”，……直到被$\sqrt{N}$的整数部分除。

如果用2除不尽，就转用3除，如不能被3除尽，就转用4除，……，直到被$\sqrt{N}$的整数部分除，如果还不能除尽，即转去打印“1*N”。

```
10  PRINT "INPUT NUMBER N=";
20  INPUT N
25  PRINT
30  IF N=0 THEN GOTO 230
40  IF SGN(N)<>-1 THEN GOTO 60
50  PRINT N;"=-1";
```

```
55  GOTO 70
60  PRINT N; "=1";
70  LET N=ABS(N)
80  FOR I=2 TO SQR(N)
100 IF N/I <>INT(N/I) THEN GOTO 150
110 LET N=N/I
130 PRINT "*"; I;
140  GOTO 100
150 NEXT I
160 IF N=1 THEN GOTO 190
180 PRINT "*"; N;
190 PRINT
200 GOTO 10
230 END
```

例6 编写一个程序，求出自然数3～100之间的所有素数。

素数又叫质数，除了1和它本身之外它不应该被任何整数整除。比如11就是素数，因为它除了1和11以外，不能被2～10之间的任何整数整除；而12不是素数，它可以被2、3、4、6整除。判断一个自然数N是不是素数，可以采用这样的方法：用从2到$\sqrt{N}$之间的所有整数去除N，如果都除不尽，则N就是素数。

```
10  FOR I=3 TO 100 STEP 2
15  LET J=INT(SQR(I))
20  FOR K=2 TO J
25  LET L=I-INT(I/K)*I
30  IF L=0 THEN GOTO 45
35  NEXT K
```

```
40  PRINT I
45  NEXT I
50  END
```

例7 验证著名的哥德巴赫猜想：任何一个大偶数都可以写成两个素数之和。下面这个程序对于4～100的每一个偶数，都打印成两个相应素数之和的形式。

```
10  FOR N=4 TO 100 STEP 2
20  FOR B=1 TO N/2 STEP 2
30  FOR C=2 TO SQR(B)
35  IF B/C=INT(B/C)THEN GOTO 75
40  NEXT C
45  LET D=N-B
50  FOR E=2 TO SQR(D)
55  IF D/E=INT(D/E) THEN GOTO 75
60  NEXT E
65  PRINT N; "="; B; "+"; D
70  GOTO 80
75  NEXT B
80  NEXT N
90  END
```

例8 编制一个统计全班同学各门课程考试成绩的程序。程序中假设全班人数不超过50人。运行后，计算机首先在屏幕上显示提问：有多少门课程？并依次请你打入各门课程名称。随后询问第一个学生的学号（NUMBER？），以及各门课程的成绩，再问第二个学生的，……。直到全班同学的成绩都打入之后，再询问学号时，回答一个0，就表示输入结束。然后计算机开始输出，先按各人平均成绩由高到低显示出全班每个

人每门课成绩和平均成绩。当你看完之后，敲一下“A”键则接着显示后面的内容。最后显示出各门课程的分数档次统计，即0～9分、10～19分、20～29分……各有多少人。由于各种普及型计算机的显示屏幕规格不大一样，读者在试验本程序时可以先用较少的人数和课程数。当人数较多、课程门数较多时需要根据具体情况适当改写一下程序中的输出部分。

```
100 PRINT "HOW MANY COURSES ARE THE-
    RE"; (有多少门课程)
105 INPUT C
110 DIM X(49, C+1), V(11, C), S$(C+1)
120 S$(0)= "NUMBER" (学号)
125 S$(C+1)= "AVERAGE" (平均成绩)
130 PRINT "PLEASE TYPE COU RSES' NAM-
    ES" (请键入课程、名字)
140 FOR I =1 TOC
142 INPUT S$(I)
144 NEXT I
146 PRINT
147 PRINT
150 FOR I=0 TO 49
152 PRINT S$(0);
154 INPUT X(I, 0)
160 IF X(I, 0)=0 THEN GOTO 290
165 LET X2=0;
170 FOR J=1 TO C
180 PRINT S$(J);
185 INPUT X(I, J);
```

```
190 LET X2=X2+X(I, J)
200 NEXT J
205 LET X2=X2/C
210 LET X(I, C+1)=INT(X2*100+0.5)/100
220 NEXT I
290 LET N=I
300 LET N1=N-1
310 FOR I=0 TO N1-1
320 FOR J=I+1 TO N1
330 IF X(I, C+1)>=X(J, C+1) THEN GOTO
    380
340 FOR K=0 TO C+1
345 LET W=X(I, K)
350 LET X(I, K)=X(J, K)
355 LET X(J, K)=W
360 NEXT K
380 NEXT J
390 NEXT I
395 PRINT S$(0);
396 FOR J=1 TO C+1
397 PRINT TAB(7+J*5); S$(J);
398 NEXT J
399 PRINT
400 FOR I=0 TO N1
405 PRINT I+1;
410 FOR J=0 TO C+1
415 PRINT TAB(5+J*5); X(I, J);
```

```
420 NEXT J
422 PRINT
425 NEXT I
426 IF KEY=65 THEN GOTO 430
```

注：若不是COMX-35计算机，此句应做适当变动

```
428 GOTO 426
430 FOR I=0 TO C
435 FOR J=0 TO N1
440 LET V(0, I)=V(0, I)+X(J, I+1)
450 LET R=INT(X(J, I+1)/10)
455 LET V(R+1, I)=V(R+1, I)+1
460 NEXT J
465 LET V(0, I)=V(0, I)/N
470 NEXT I
480 FOR I=1 TO C+1
482 PRINT TAB(I*8); S$(I);
484 NEXT I
490 PRINT
500 PRINT
510 FOR I=1 TO C+1
512 PRINT TAB(I*8); INT(V (0, I-1)*10+0.5)
    /10;
514 NEXT I
516 PRINT
520 FOR I=1 TO 10
522 PRINT(I-1)*10; "--"; (I-1)*10+9
530 FOR J=1 TO C+1
```

```
532 PRINT TAB(J*8); V(I, J-1);
534 NEXT J
536 PRINT
540 NEXT I
550 PRINT "  "; 100
560 FOR J=1 TO C+1
565 PRINT TAB(J*8); V(11, J-1);
566 NEXT J
568 PRINT
600 END
```

例9　编写一个BASIC程序，打印1984年日历表。

我们知道，84年是闰年，2月有29天，而且元旦那一天是星期日。

```
10  PRINT, "1984"
20  PRINT, "= = = = "
30  PRINT "␣␣␣␣␣0␣␣␣␣␣1␣␣␣␣␣2
    ␣␣␣␣␣3␣␣␣␣␣4␣␣␣␣␣5␣␣␣␣␣6"
40  PRINT "␣␣␣␣SUN␣␣MON␣␣TUE
    ␣␣WED␣␣THU␣␣FRI␣␣SAT"
50  PRINT
60  LET K=0
70  FOR I=1 TO 12
80  PRINT, I; "(MONTH)"
90  READ N
100 FOR J=1 TO N
110 PRINT TAB(4+7*K); J;
120 LET K=K+1
```

```
130 IF K<=6 THEN GOTO 160
140 PRINT
150 LET K=0
160 NEXT J
170 PRINT
180 PRINT
190 NEXT I
200 DATA 31, 29, 31, 30, 31, 30, 31, 31, 30,
    31, 30, 31
220 END
```

例10　100! 的末尾有多少个零?

所谓100! 就是从1到100共一百个正整数连乘。分析一下便可知道，这个乘积的因数中有多少个5，积的末尾就有多少个零。

```
 3  LET K=0
 5  FOR I=5 TO 100 STEP 5
10  LET J=I/5
15  IF J<>INT(J) THEN GOTO 35
20  LET K=K+1
25  LET J=J/5
30  GOTO 15
35  NEXT I
40  PRINT K
45  END
```

例11　编写计算组合数C_n^m的BASIC程序。

在高中代数中学过排列组合的知识，可知

$$C_m^n = \frac{m!}{n!(m-n)!},\quad C_m^0 = 1,\quad (n \leqslant m)$$

可见，计算C_m^n要多次用到阶乘计算。因此，把计算阶乘编成子程序。

```
10  PRINT "VALUES OF M, N=";
20  INPUT M,N
30  PRINT
40  LET K=M
50  GOSUB 300
60  LET C=F
70  LET K=N
80  GOSUB 300
90  LET C=C/F
100 LET K=M-N
110 GOSUB  300
120 PRINT "C(M,N)=";  C/F
130 END
300 LET F=1
310 FOR I=1 TO K
320 LET F=F*I
330 NEXT I
340 RETURN
```

例12　小李每次上楼梯，有时一步迈一阶，有时一步迈两阶。现在要到二十阶上面的楼上去，问一问小李有多少种上楼的方法？

上楼梯第一级，只有一种方法；上楼梯的第二级就有两种方法了。一般地说，若想上第n级楼梯，因为他的最后一步可

以从第n－2级跨两阶，也可以从第n－1级跨一级而达到。所以若用u_{n-2}，u_{n-1}、u_n分别表示上第n－2，n－1和第n级楼梯的方法数，便可得下列关系式：

$$u_n = u_{n-1} + u_{n-2}$$

由此可编出程序：

```
10  LET A=1
15  LET B=2
20  FOR I=1 TO9
25  LET A=A+B
30  LET B=A+B
35  NEXT I
40  PRINT B
45  END
```

例13 试为我国古代著名的孙子定理编一个BASIC程序。孙子定理也称为中国余数定理，又叫韩信点兵。原问题用现代语言叙述就是：有一个数，用3除余2，用5除余3，用7除余2，求此数。分析题意可以知道，这个问题的解法之一是：先把这个数定为2，然后每次加一个3，直到所得的数被5除能余3时为止；再在这个数的基础上每次加15，直到所得数被7除能余2为止。最后所得的这个数就是满足原题条件的最小的一个数。

```
10  READ I,X
20  READ J,K
30  LET X=X+I
40  IF (X-J*INT(X/J))=K THEN GOTO 60
50  GOTO 30
60  IF J=7 THEN GOTO 90
```

```
 70  IF I/J < > INT (I/J) THEN LET I=I*J
 80  GOTO 20
 90  PRLNT "X=" ; X
100  DATA 3, 2, 5, 3, 7,2
120  END
```

这个程序可以用来解类似的问题。比如：在你面前有一条长长的楼梯，若你每步跨2阶，那末最后剩下1阶；若你每步跨3阶，那么最后剩下2阶；若你每步跨5阶，最后剩4阶；若每步跨6阶，最后剩5阶；只有当你每步跨7阶时，最后才正好走完，一阶也不剩。问一问这条楼梯至少有多少阶？解这个问题仍用上面那个程序，第60句仍为IF J=7 THEN GOTO 90，但第100句改为：100 DATA 2，1，3，2，5，4，6，5，7，0就行了。

例14 最后，请你看一个物理学中平抛物体运动的示教、练习程序。它是在Apple-II plus计算机上实现的，利用了Apple机所具有的高分辨率绘图功能。

当一个物体从一个桌面上以一定的初速度V_0沿水平方向抛出去，如果不计空气阻力，物体离开桌面后将沿着一条抛物线轨迹作曲线运动。这个运动可以看作是两个分运动的合成：一个是水平方向的匀速直接运动，另一个是竖直方向的自由落体运动。设桌面高为H，取水平方向为x轴，竖直方向为y轴，桌脚处为原点。以物体离开桌面的时刻为t=0。此时物体的座标为：x=0，y=H。

任一时刻t，物体的座标为：$x=V_0t$，$y=H-\frac{1}{2}gt^2=H-\frac{gx^2}{2V_0^2}$。物体落地时刻记为$t_M$此时，$x=x_M=V_0t_M$，$y=0$。可以

推导出：$t_M=\sqrt{\frac{2H}{g}}$，$x_M=V_0\sqrt{\frac{2H}{g}}$。

这个程序运行之后，首先画出一个桌面和在桌面边沿处的物体。然后画出这个物体作平抛运动的一条轨迹，并分步画出水平、竖直两个分运动的轨迹。在屏幕下方列出几个主要公式。稍待片刻就又画出另外两条平抛轨迹，相当于初速度更大和更小一些的情况。最后询问你是否希望进行计算，有两个计算模式：①已知桌高H和初速度V_0，求落地点的横座标X_M；②已知H和X_M，求初速V_0。于是你就可以反复进行练习了。

下面就是这个程序，其中10～70行是画出桌面和物体，80行是画平抛轨迹，100～140行是画出两个分运动的轨迹。150～180行显示计算公式。200～220行分别画出另外两条平抛运动轨迹。250～300行为计算X_M或V_0。

```
10  HGR: HCOLOR= 3
20  HPLOT 5,45
25  FOR I= 6 TO 100: HPLOT TO I,45:
    NEXT I
30  FOR J=46 TO 50: HPLOT TO 100,J:
    NEXT J
40  FOR I=100 TO 95 STEP - 1 : HPLOT
    TO I, 50 : NEXT I
50  FOR J=51 TO 150 : HPLOT TO 95, J :
    NEXT J
60  FOR I= 95 TO 270 : HPLOT TO I, 150
    : NEXT I
70  HPLOT 100, 40 TO 96, 36 TO 92, 40 TO
    96, 44 TO 100, 40
```

```
 80  FOR  I= 1  TO 150 : HPLOT TO (I+100),
     (40+I/150*I/150*110) : NEXT I
100  HCOLOR= 2
110  FOR I= 0  TO  4
115  F=100+30*I : F1=F+30
120  HPLOT F, 40 TO F1, 40
125  HPLOT TO F1, 30
130  H=40+I*30*I*30*110/22500 : H1=40+
     (I+1)*(I+1)*900/22500*110 : HPLOT
     100,H TO 100,H1
135  HPLOT 100,H1 TO 92, H1
138  FOR D=1 TO 5000:NEXT D
140  NEXT I
150  PRINT "X=V*T"
160  PRINT "Y=H-G*T^2/2"
170  PRINT "Y=H-G*X^2/(2*V^2)"
180  PRINT "XM=V*SQR(2*H/G) (Y=0)"
200  HCOLOR=5:HPLOT 100,40
210  FOR I=1 TO 170:HPLOT TO (I+100),
     (40+I/170*I/170*110): NEXT I
215  HCOLOR=6 : HPLOT 100,40
220  FOR I=1 TO 100 : HPLOT TO (I+100),
     (40+I/100*I/100*110) : NEXT I
250  PRINT"DO YOU WANT TO CALCULATE ";
     " ; : INPUT A$
260  IF LEFT$(A$,1)<> "Y" THEN 300
270  PRINT" 1. H,V ARE KNOWN,XM=? "
```

```
272  PRINT "2. H, XM ARE KNOWN,V = ? "
274  PRINT "0. EXIT."
276  PRINT "WHICH" ; : INPUT E
280  ON(E+1) GOTO 300,285,290
285  INPUT "H(CM),V(CM/S) = " ;H,V:PRINT
     "XM = ; V * SQR(2 * H/980): GOTO 270
290   INPUT" H(CM),XM(CM) = " ; H,XM
    :PRINT "V = " ; XM/SQR(2 * H/980):GOTO
    270
300  END
```

由于篇幅所限，我们不再多举例子了。现在怎么样？你是不是跃跃欲试、打算自己动手编些实际应用程序？大胆地去实践吧！世上无难事，只怕有心人，入门不难，深造也是办得到的。多学多练，你就会成为“计算机应用”小行家。在这条道路上，如果这本小册子能对你有初步的帮助，那我们将是非常高兴的。

附录1　　Microsoft BASIC 语言一览表

运算符

算术运算符　+，－，＊，／，↑

关系运算符　＞，＜，＝，＞＝，＜＝，＜＞

逻辑运算符　AND，OR，NOT

函数

类型	函数名	含义	备注
算术函数	SQR（参量）	平方根	参量≥0
	INT（参量）	取整数部分	-10E38≤参量≤10E38
	RND（参量）	产生随机数	参量≥0
	ABS（参量）	取绝对值	
	SGN（参量）	参量的符号	
	COS（参量）	余弦函数	参量单位：弧度
			-9999999≤参量≤9999999
	SIN（参量）	正弦函数	同上
	TAN（参量）	正切函数	同上
	ATN（参量）	反正切函数	函数单位：弧度
			-10E38≤参量≤10E38
	EXP（参量）	指数函数（e=2.71823）	-10E38≤参量≤37
	LOG（参量）	自然对数	参量＞0
字符串函数	LEN（字串参量）	字符串的长度	
	STR$（参量）	把数字参量变为字符串	
	VAL（字串参量）	字符串参量的数字值	字符串参量的内容必须是数字字串
	ASC（字串参量）	字符串参量第一个字符的ASC码（十进制）	
	CHR$（参量）	以参量作为ASC码所代表的符号	0≤参量≤255

续

类型	函数名	含义	备注
字符串函数	LEFT$（字串参量1，参量2）	字符串参量1中左边若干个字符，个数由参量2决定。	
	RIGHT$（字串参量1，参量2）	字符串参量1中右边若干个字符，个数由参量2决定。	
	MID$(字串参量，参量1，参量2)	字符串参量中间若干个字符，个数由参量2决定。起始位置由参量1决定——自第某个字符开始。	
	INKEY$	检查键盘，看看刚打入的是个什么字符。	在程序中一般使用形式为：LET串变量=INKEY$
绘图函数	POINT（参量1，参量2）	分别以参量1，2为列号、行号的那个点的颜色码	0≤参量1≤127 0≤参量≤63
	PEEK（参量）	某个存贮单元中所存的十进制数值，该单元的地址由参量决定	-32768≤参量≤32768
	INP（参量）	从指定的一个输入输出通路中读取一个值。	通路号： 0≤参量≤255 在程序中一般使用形式为：LET变量=INP（参量）
	USR（参量）	调用用户的汇编语言子程序	在程序中一般使用形式为：LET变量=USR（参量）

语句

形　式	功　能	备　注
DIM变量名（参量）或 DIM字串变量名(参量)	指定数组大小	最多到三维数组
STOP	停止程序运行	
END	终止程序	
GOTO行号	跳到指定的那一行去继续运行。	
GOSUB行号	转到子程序	
RETURN	子程序结束	
FOR变量=参量1TO 参量2STEP参量3	实行循环	参量3≠0
NEXT		
REM	注释	
CLEAR参量	清变量空间	
IF表达式1THEN表达式2ELSE表达式3	条件分支	表达式1也可以是逻辑表达式，表达式2、3也可以是行号
INPUT字串常量 变量1〔变量1$〕，……	把字串常量打印出来并询问应当把什么值赋给变量1。	
INPUT#“文件名” 变量1〔变量1$〕，……	从指定的数据文件中读取变量的值。	
PRINT参量1〔字串型参量〕，……	在屏幕上显示参量。	使用；或，号
PRINT#“文件名” 变量1〔变量1$〕，……	把变量内容送到指定的数据文件	
PRINT TAB（参量1） 参量2〔字串参量2〕	与PRINT类似，但有若干个空格，个数由参量1决定。	0≤参量1≤255
PRINT USING字串； 参量1，……	与PRINT类似，但按指定格式输出。	
PRINT @ 参量1，参量2〔字串参量2〕，……	与PRINT类似，在指定位置（参量1）上开始打	0≤参量1≤511

形　式	功　能	备　注
	印参量2。	
LET 变量＝数字表达式 或LET 字串变量＝字串表达式	赋值。	
DATA常量1，……	为READ语句指定值。	
READ 变量1，……	把DATA语句中的值赋给变量1……。	
RESTORE	恢复数据区。	
CLS	在绘图状态中清除屏幕	
COLOR 参量1，参量2	设定颜色	参量 2 ＝ 0 或 1 在绘图状态： 1≤参量1≤ 4 在文字状态： 1 ≤参量 1 ≤ 8
SET（参量1，参量2）	在指定的列号行号处画上一点	0 ≤参量1≤127 0 ≤参量2≤63
RESET(参量1，参量2)	在指定的列号行号处抹去一点	同上
MODE（参量）	选择绘图状态或文字状态	参量 ＝ 0 或 1
SOUND参量1，参量2	产生指定频率和持续时间的声音	0 ≤参量1≤31 1 ≤参量2≤9
POKE参量1，参量2	把一个数（参数 2）存入指定的内存单位（地址为参量 1）	0 ≤参量2≤255 －32768≤参量1 ≤32767
LPRINT参量 1 〔字串参量 1〕，……	与 PRINT 类似，但在印字机上打印	
OUT（参量1，参量2）	把参量 2 值送到一个输入输出通路去（其号数为参量 1）。	0 ≤参量1，2≤255
COPY	把屏幕上的内容用印字机复印出来。	

命令

形　式	功　能	备　注
LIST　行号1—行号2 或：LIST LIST行号1— LIST—行号2	列出程序行	行号2≥行号1
LLIST	与LIST类似，但在印字机上打印。	
NEW	清除屏幕和内存的内容。	
RUN或RUN行号	开始执行程序	
CRUN“文件名”	从磁带中装入一个程序并执行之	文件名仅头16个字符有效。
CONT	继续程序的执行	
CLOAD“文件名”	从磁带中装入一个程序。	文件名仅头16个字符有效。
CSAVE“文件名”	向磁带中存入一个程序文件。	同上。
VERIFY“文件名”	把磁带中的程序和内存中的程序进行比较，检验它们是否相同。	同上。
BREAK（即CTRL—）	中止程序	

附录2 字符——代码对照表
(参照美国标准信息交换代码ASCII)

十进制码	八进制码	字　符
0	000	NULL
1	001	CTRL—A
2	002	CTRL—B
3	003	CTRL—C
4	004	CTRL—D
5	005	CTRL—E
6	006	CTRL—F
7	007	CTRL—G (响)
8	010	CTRL—H
9	011	CTRL—I
10	012	CTRL—J (换行)
11	013	CTRL—K
12	014	CTRL—L
13	015	CTRL—M (回车)
14	016	CTRL—N
15	017	CTRL—O
16	020	CTRL—P
17	021	CTRL—Q
18	022	CTRL—R
19	023	CTRL—S
20	024	CTRL—T
21	025	CTRL—U
22	026	CTRL—V
23	027	CTRL—W
24	030	CTRL—X
25	031	CTRL—Y
26	032	CTRL—Z
27	033	ESC
28	034	FS

续

十进制码	八进制码	字　符
29	035	GS
30	036	RS
31	037	US
32	040	SPACE　(空格)
33	041	!
34	042	"
35	043	#
36	044	$
37	045	%
38	046	&
39	047	'　(重音号)
40	050	(
41	051	)
42	052	*
43	053	+
44	054	,
45	055	—
46	056	.
47	057	/
48	060	0
49	061	1
50	062	2
51	063	3
52	064	4
53	065	5
54	066	6
55	067	7
56	070	8
57	071	9
58	072	:
59	073	;
60	074	<

续

十进制码	八进制码	字　符
61	075	=
62	076	>
63	077	?
64	100	@
65	101	A
66	102	B
67	103	C
68	104	D
69	105	E
70	106	F
71	107	G
72	110	H
73	111	I
74	112	J
75	113	K
76	114	L
77	115	M
78	116	N
79	117	O
80	120	P
81	121	Q
82	122	R
83	123	S
84	124	T
85	125	U
86	126	V
87	127	W
88	130	X
89	131	Y
90	132	Z
91	133	[
92	134	\

续

十进制码	八进制码	字符
93	135	]
94	136	↑
95	137	←
96	140	`
97	141	a
98	142	b
99	143	c
100	144	d
101	145	e
102	146	f
103	147	g
104	150	h
105	151	i
106	152	j
107	153	k
108	154	l
109	155	m
110	156	n
111	157	o
112	160	p
113	161	q
114	162	r
115	163	s
116	164	t
117	165	u
118	166	v
119	167	w
120	170	x
121	171	y
122	172	z
123	173	{
124	174	/

续

十进制码	八进制码	字　符
125	175	}
126	176	~
127	177	DEL

附录3 错误信息表

(a) LASER-310机错误信息表

序号	显示	说明
1	BAD FILE DATA	用PRINT#存到磁带上的数据的类型与用INPUT#读出的变量类型不一致
2	CAN'T CONT	当：1）由于错误而中断运行，2）程序在中断过程中进行了修改，或3）并不存在时，仍试图去继续运行程序
3	DISK COMMAND	没有联结磁盘驱动器却企图使用磁盘命令
4	DIVISION BY ZERO	在一个表达式中遇到用0作除数，或是逐渐展开的运算导致于0的负方幂
5	ILLEGAL DIRECT	语句不能以立即方式加以执行，例如：INPUT
6	FUNCTION CODE	算术函数、字符串函数后面的参量超出范围。在下列情况，也会出现这个错误：1）下标过大或为负数，2）对负数或0求对数，3）负数求平方根，4）乘方运算中底数为负而指数为非整数，5）调用USR函数却没有给起始地址，6）MID$、LEFT$、RIGHT$、INP、OUT、PEEK、POKE或TAB中参量不合适
7	LOADING ERROR	装入计算机的程序有错误
8	MISSING OPERAND	某些命令缺少被操作数
9	NEXT WITHOUT FOR	一个NEXT语句中的循环变量找不到前面相应的FOR语句
10	OUT OF DATA	执行READ语句时程序中已经没有未被读过的DATA语句了
11	OUT OF MEMORY	程序太大，FOR-NEXT循环或转子太

		多，变量太多、复杂的表达式太多。
12	OUT OF SPACE	串变量引起BASIC超过未用的存贮器容量
13	OVERFLOW	计算结果用数的形式表示起来太大。如果是下溢出发生，则把计算结果视为0并继续执行，好象没有错误一样。
14	REDIM'D ARRAY	对同一个数组给了两个DIM语句，或是一个数组原已隐含为10个变量以后又给一个DIM语句。
15	REDO	在执行一个INPUT命令时把一个字符串赋给一个数字型变量。
16	RET'N WITHOU-T GOSUB	遇到一个RETURN语句，在此之前却没有相应的GOSUB语句。
17	FORMULA TOO COMPLEX	一个字串表达式太长或太复杂。应把它分为几个小一些的表达式
18	STRING TOO LONG	试图产生一个长度超过255个字符的字符串
19	BAD SUBSCRIPT	某个数组元素的下标超出定义的范围或下标值错误
20	SYNTAX	遇到一行程序其中包含有错误的字符顺序（比如圆括弧不配对，标点符号错误，命令或语句拼错等）
21	TYPE MISMATCH	把一个数字值赋给一个字符串变量，或是反过来；一个需要数字型参量的函数却被送一个字符串型参量，或是反过来
22	UNDEF'D STATE-MENT	在GOTO、GOSUB或IF…THEN……ELSE语句中涉及到某一行号，但实际程序中并不存在那样一行
23	VERIFY ERROR	磁带中的程序与内存中的程序不一样

(b) COMX-35机错误信息表

00 由用户引起的程序中断
01 在ASC或LEN函数中有语法错误
02 数组超出范围或无定义
03 维数错误
04 DEFINT有违反规定的结尾
05 在自变量中缺少括号
06 自变量超出范围
07 遇到混合型式的计算
08 以0为底数的错误，或负数的对数
09 遇到不可执行的函数
10 EXIT指令必须与FOR/NEXT或GOSUB/RETURN一起使用
11 FOR-NEXT堆栈溢出，或FOR/NEXT直接执行
12 FOR中有语法错误
13 GOSUB堆栈溢出
14 在十六进制数中有不可接受的字符
15 浮点数太大以致转换成整数或整数乘溢出
16 在条件语句中有不可接受的算符
17 INPUT或FVAL不能直接执行
18 在READ中必须有变量名或字符串名
19 在READ或INPUT中有语法错误
20 在LEN函数中有语法错误
21 在赋值语句中有语法错误
22 缺少引号
23 在LIST中有语法错误
24 在库中找不到这样的字
25 在MID$函数中有语法错误
26 在NEXT语句中发现有不可接受的变量名

27 应该不是数就是字
28 缺少运算括号
29 在POKE语句中自变量的个数错误
30 在PRINT语句中最后的字符不可接受
31 在DATA语句中有语法错误
32 数据不够
33 在INPUT语句中找不到这样的字符串
34 在赋值语句中缺少等号
35 在字符串数组中缺括号
36 在USR或CALL中自变量太多
37 在CHR$函数中有语法错误
38 在二进制数中有不可接受的字符
39 行缓冲器溢出
40 输入时文件未打开
41 输出时文件未打开
42 行的结尾不可接受或语句不能执行
43 堆栈溢出
44 数的位数太多
45 在数中有不可接受的字符
46 找不到这样的行号
47 在IF语句中有不可接受的运算
48 存贮器溢出
49 在MOD语句中自变量的个数错误
50 程序太长存贮器不够存贮
51 自变量超限
52 自变量的个数错误
53 自变量的个数错误
54 字符串变量未定义
55 磁带读错
56 磁带写错

57 文件不是BASIC程序
58 文件不是BASIC数据
59 被保留的
60 被保留的
61 被保留的
62 未送ROM或ROM卡片
63 没有足够的存贮单元供重新编号
64 重编号设置的行号有错
65 没有定义这样的下标变量
66 字符串超出127个字符
67 在COLOR中自变量的个数必须是1
68 在SCREEN中自变量的个数必须是1
69 在CTONE中自变量的个数必须是1
70 在VOLUME中自变量的个数必须是1
71 在NOISE中自变量的个数必须是2
72 在TONE中自变量的个数必须是3

附录4 普及型微计算机常用名词中英文对照表

assembler 汇编程序

binary numbers 二进制数
bit (二进制) 位
byte 字节，位组
cassette 盒式磁带机
compiler 编译程序
control unit 控制器
CPU(Central Processing Unit) 中央处理机
CRT(Cathode Ray Tube) 阴极射线管
data 数据
decimal numbers 十进制数
EAR 耳机
floppy disk 软磁盘
flowchart 流程图
graphic display 图形显示
instruction 指令
interface 接口
interpreter 解释程序
keyboard 键盘
light pen 光笔
LSI (Large Scale Intergration) 大规模集成电路
machine cycle 机器周期
machine language 机器语言
memory 存贮器
microcomputer 微型计算机
microprocessor 微处理器

monitor 监视器
MIC 传声器，话筒
operators 运算符
object program 目标程序
printer 打印机
program 程序
power ON 电源开
OFF 关
record 记录
RAM (Random Access Memory) 随机存取存贮器
ROM (Read Only Memory) 只读存贮器
screen 荧光屏
source program 源程序
string 字符串
teletype 电传打字机
TV 电视，电视接收机
word 字

果粉藏書

参 考 书 目

① 谭浩强、田淑清、谢锡迎：《BASIC语言》，科学普及出版社，1981。

② 华东师范大学计算机科学系、郑州工学院科研所：《BASIC程序设计和算法基础》，1983。

③ 谭浩强、王攻本：《计算机基础教程》，清华大学出版社，1984。

④ 《COMX-35微型计算机使用手册》

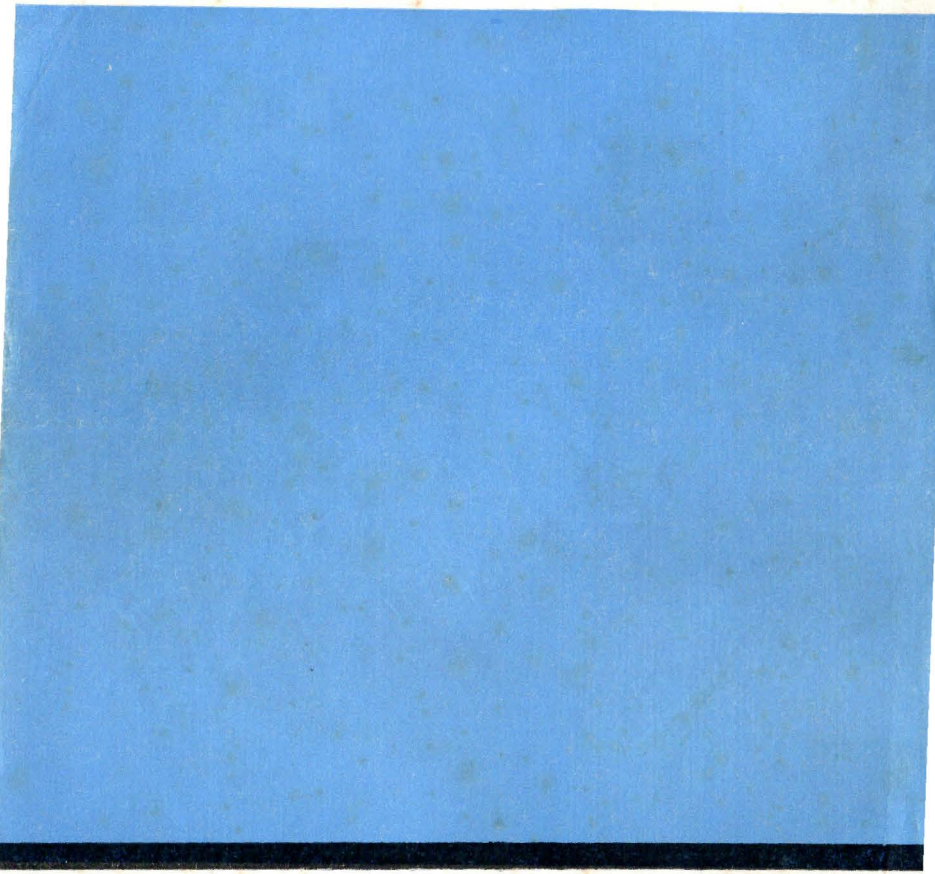
⑤ 《LASER-310微型计算机使用手册》

⑥ 周明德：《微型计算机硬件软件及其应用》，清华大学出版社，1983。

⑦ 〔美〕亚当·奥斯本：《微型计算机入门 初学者的书》人民邮电出版社，1980。

〔美〕亚当·奥斯本：《微型计算机入门 基本概念》人民邮电出版社，1981。

封面设计：邵　新



科技新书目：91—124·统一书号：15045·总3043—无6334·定价：0.95元